SAAD DAHLAB UNIVERSITY OF BLIDA 1

Faculty of Sciences

Department of Computer Science

Master's Thesis

Speciality: Software Engineering

THEME :

Leveraging Knowledge Graphs and Advanced AI for Medical Interpretation

Presented by :

Abdelmadjid Maallemi

Riadh Hadjallah Hedjalla

Supervised by:

Dr. M. FAREH

Name. Dr. M. MEZZI

Name. Dr. N. LAHIANI

june 2025

Acknowledgements

First and foremost, we praise and thank Allah Almighty for blessing us with health and strength, for granting us the willpower and perseverance to work hard, and for surrounding us with sincere and supportive people whose presence played a vital role in the completion of this work.

We extend our deepest gratitude and appreciation to **Ms. Fareh** for her continuous support, her trust in us, and her valuable assistance throughout our journey with this project. She has been by our side from the very beginning and never hesitated to provide us with her insightful advice and guidance, which greatly contributed to the successful completion of our work. It was a great honor to work with her, and we have benefited immensely from her experience and dedication.

We would also like to express our sincere thanks to everyone who contributed, directly or indirectly, to the realization of this thesis. We particularly acknowledge *Dr. Hadjallah yasmine*, General Practitioner, for her support and expertise, which were of great value to our work.

Finally, we extend our heartfelt gratitude to our beloved families: our parents, brothers, sisters, and friends, for their unwavering support, encouragement, and care throughout the preparation of this thesis. Their presence by our side was a priceless source of strength and comfort.

Dedications:

I dedicate this memoire:

To my **mother**, Who stood by my side unconditionally, supporting me with all her strength through good times and bad, in exhaustion and in rest, in moments of despair and hope. Her prayers lit my path... I dedicate this work as a token of deep appreciation for her unmatched love and her countless sacrifices.

To my dear father, **Mohamed**, The reason for my existence and my progress, my first role model and strongest supporter, who spared neither effort nor time nor resources to help me reach this stage. He stood by me with love and selflessness. I dedicate this thesis in recognition of your kindness and with heartfelt gratitude for all you have done.

To my little brother, **Chakib**, My companion on this journey and my friend who never left my side. You were always there for me, sharing moments of joy and hardship, offering me peace with your simple words and constant presence. You were more than a brother—you were comfort in tough times and joy in moments of success. I dedicate this work to you with love and gratitude for your pure soul, which gave me so much without asking for anything in return.

To my two little sisters, **Chiraz** and **Malak**, My eternal joy, the heartbeat of my life, and the most precious treasures I have. To you, who fill my days with love, laughter, and innocence, and who unknowingly give me the energy to be stronger and better. Your presence in my life is a blessing beyond measure, and your voices and laughter have always been a soothing balm to my exhaustion. I dedicate this work to both of you with love and in recognition of the special place you hold in my heart.

To my colleague and journey companion, **Riyad**, Who shared this path with me from beginning to end, through five years of hard work, challenges, and dedication. We were more than just classmates—we were partners at every step, and your presence was a true source of support in every stage of this journey. You have become one of my dearest friends. I thank you for your precious friendship and for everything you have contributed throughout these years. I sincerely wish you all the best and continued success in your life and career.

Abdelmadjid Maallemi

Dedications:

I would like to sincerely thank my family and friends for their unwavering support and encouragement throughout this journey.

To my beloved father and mother—your sacrifices, your patience, and your unconditional love have been the foundation of everything I've achieved. Your guidance and belief in me gave me the strength to push forward, even in the most challenging moments.

Thank you for always being there, for your endless encouragement, and for helping me become the person I am today.

This accomplishment is as much yours as it is mine.

Riadh Hadjallah Hedjalla.

Abstract

Knowledge graphs are graph-based structures that represent entities and relationships as triples, enabling machines to perform semantic reasoning. They are widely used in fields such as healthcare. This project aims to build a medical diagnostic system based on a knowledge graph. The system integrates explainable deep learning models and leverages knowledge graph embeddings to enhance interpretability. The design follows multiple stages: collecting and preprocessing data, transforming it into RDF format, and constructing a knowledge graph that captures patients, symptoms, risk factors, and diseases.

Embeddings are generated using Node2Vec, converting graph nodes into numerical vectors suitable for deep learning models. A Graph Attention Network (GAT) is trained for diagnosis, and SHAP (SHapley Additive exPlanations) and LIME (Local Interpretable Model-agnostic Explanations) are used to interpret model results both globally and locally, identifying important features. The study is conducted on lung cancer diagnosis, and the obtained results are interpreted using explainable AI techniques. The embeddings and model are evaluated using accuracy, F1-score, precision, and recall. The combination of GAT with Node2Vec outperforms all other configurations and is selected for final deployment. The proposed solution demonstrates strong predictive performance and high interpretability, aligning AI decisions with medical reasoning.

Key words:

Knowledge Graph, Node2Vec, Graph Embedding, GAT (Graph Attention Network), Explainable AI (XAI), SHAP, LIME

Résumé

Les graphes de connaissances sont des structures basées sur des graphes qui représentent des entités et des relations sous forme de triplets, permettant aux machines de réaliser un raisonnement sémantique. Ils sont largement utilisés dans des domaines comme la santé. Ce projet vise à construire un système de diagnostic médical basé sur un graphe de connaissances. Le système intègre des modèles d'apprentissage profond explicables et exploite les embeddings de graphes de connaissances pour améliorer l'interprétabilité. La conception suit plusieurs étapes : collecte et prétraitement des données, transformation au format RDF, puis construction d'un graphe de connaissances représentant les patients, les symptômes, les facteurs de risque et les maladies.

Les embeddings sont générés à l'aide de Node2Vec, transformant les nœuds du graphe en vecteurs numériques adaptés aux modèles d'apprentissage profond. Un réseau de neurones à attention sur graphe (GAT) est entraîné pour le diagnostic, et les méthodes SHAP (SHapley Additive exPlanations) et LIME (Local Interpretable Model-agnostic Explanations) sont utilisées pour interpréter les résultats du modèle à la fois globalement et localement, en identifiant les caractéristiques importantes. L'étude est appliquée au diagnostic du cancer du poumon, et les résultats obtenus sont interprétés grâce aux techniques d'intelligence artificielle explicable. Les embeddings et le modèle sont évalués à l'aide de la précision, du score F1, de la précision (precision) et du rappel (recall). La combinaison de GAT avec Node2Vec surpasse toutes les autres configurations et est sélectionnée pour le déploiement final. La solution proposée démontre une forte performance prédictive et une grande interprétabilité, alignant les décisions de l'IA avec le raisonnement médical.

Mots clés:

Graphe de connaissances, Node2Vec, embedding de graphe, GAT (Graph Attention Network), intelligence artificielle explicable (XAI), SHAP, LIME.

الملخص

تُعد الرسوم المعرفية هياكل تعتمد على الرسوم البيانية لتمثيل الكيانات والعلاقات على شكل ثلاثيات، مما يُمكّن الآلات من إجراء استدلال دلالي. وتُستخدم على نطاق واسع في مجالات مثل الرعاية الصحية. يهدف هذا المشروع إلى بناء نظام تشخيص طبي قائم على الرسم المعرفي. حيث يدمج هذا النظام نماذج تعلم عميق قابلة للتفسير ويستفيد من تضمين الرسوم المعرفية لتعزيز قابلية الفهم. يتبع تصميم النظام مراحل متعددة تشمل جمع البيانات ومعالجتها، وتحويلها إلى صيغة RDF، وبناء رسم معرفي يُجسّد معلومات عن المرضى والأعراض وعوامل الخطر والأمراض.

يتم إنشاء التضمينات باستخدام تقنية Node2Vec، حيث تُحوَّل عقد الرسم البياني إلى متجهات رقمية مناسبة لنماذج التعلم العميق. يتم تدريب شبكة انتباه بياني (GAT) لأغراض التشخيص، كما تُستخدم تقنيتا SHAP وLIME لتفسير نتائج النموذج على المستويين العام والمحلي، مما يُسهم في تحديد الخصائص المهمة. وقد تم تطبيق الدراسة على تشخيص سرطان الرئة، وتم تفسير النتائج المحصل عليها باستخدام تقنيات الذكاء الاصطناعي القابل للتفسير. تُقيَّم التضمينات والنموذج باستخدام مقاييس مثل الدقة، ودرجة F1، والتذكُّر، والاسترجاع. وقد أظهرت مجموعة GAT مع Node2Vec أداءً أفضل من جميع التكوينات الأخرى، وتم اختيارها للنشر النهائي. يُظهر الحل المقترح أداءً تنبؤياً قوياً وقابلية تفسير عالية، مما يربط قرارات الذكاء الاصطناعي بالمنطق الطبي.

الكلمات المفتاحية:

الرسم المعرفي، Node2Vec، تضمين الرسم البياني، شبكة الانتباه البياني (GAT)، الذكاء الاصطناعي القابل للتفسير (XAI)، LIME، SHAP.

Contents

List of Figures

List of Tables

General Introduction

Context

The field of medical diagnosis is undergoing a major transformation due to the rapid advancement of artificial intelligence technologies, which are increasingly being used to support doctors in making accurate and effective decisions. Among the most impactful tools in this domain is the **Knowledge Graph**, a graph-based model used to represent knowledge in a structured and interconnected manner. In this model, nodes represent entities (such as diseases, symptoms, medications), while edges represent the relationships between them. This structure enables a deeper understanding of the medical context.

With the evolution of deep learning algorithms, a new challenge has emerged: explaining the results generated by Deep learning models. In medical applications, it is not enough for recommendations to be accurate—they must also be understandable and interpretable by healthcare professionals. At this point, **Explainable AI** (XAI) becomes essential—a branch of artificial intelligence dedicated to developing models that offer transparent outcomes, accompanied by logical justifications for their decisions. Such capabilities allow doctors to understand how a system arrived at a certain outcome, thus reinforcing trust in AI-assisted clinical practices.

Problem Statement

With the rapid spread of many chronic and infectious diseases, the accuracy and reliability of medical diagnosis has become a central challenge in the healthcare sector. An incorrect or poorly justified medical decision can lead to serious consequences, including deterioration of the patient's condition or inappropriate treatment. As such, building trust in AI-assisted diagnosis systems is no longer optional, but essential.

However, one of the major obstacles undermining this trust is that many modern AI systems especially those based on deep learning lack interpretability. In the medical field,

it is not sufficient to produce accurate results; those results must also be supported by clear explanations that healthcare professionals can understand and evaluate. This highlights the importance of Explainable AI, which aims to close the gap between model performance and user interpretability, making AI tools more acceptable for real clinical environments.

In this context, a number of research questions arise that highlight the challenges associated with integrating explainable artificial intelligence with knowledge graph technologies and medical data analysis, including:

1. How can medical knowledge and the relationships between medical entities be represented in order to construct a knowledge graph ?
2. Which deep learning models are most suitable for diagnosis using knowledge graphs, and how can they be integrated to provide reliable and interpretable medical diagnoses?
3. Which Explainable Artificial Intelligence techniques can be applied in the medical context to enhance the transparency and interpretability of model outcomes, and how can they be used in conjunction with knowledge graphs ?

Objectives

The goal of this work is to develop a **medical diagnostic system capable of providing transparent explanations** for its decisions. To achieve this overall aim, the following specific objectives have been defined:

- **Collect and analyze medical data**: The first objective involves gathering a set of relevant medical data (such as symptoms, diseases, test results, and treatments) from reliable sources, and preparing it for knowledge graph construction.
- **Represent the data in a Knowledge Graph:** This step aims to build a structured and interconnected knowledge graph representing the medical information, enabling a deeper understanding of the relationships between medical entities and enhancing inference capabilities.
- **Develop a dedicated model for medical diagnosis:** Create and train a deep learning model capable of processing the knowledge graph data to support diagnostic tasks.
- **Apply Explainable AI techniques:** Integrate interpretability methods into the developed model to generate clear and understandable justifications for its diagnostic decisions.

Thesis Structure

In order to achieve the objectives of our work, the structure of this thesis is organized as follows:

— Chapter 1: KNOWLEDGE GRAPHS AND EXPLAINABLE AI
In this chapter, we presented the theoretical framework upon which our project is based. We began with an introduction to knowledge graphs, tracing their origins and historical development, before highlighting their significance. We also provided concrete examples of their use, while exploring their various fields of application—particularly in the medical domain. Subsequently, we discussed explainable artificial intelligence by presenting its main techniques. The chapter concluded with a state-of-the-art review, in which we analyzed several research works related to our topic. This analysis enabled us to identify existing gaps and guided the direction of our project to address these shortcomings while leveraging identified opportunities.

— Chapter 2: SYSTEM DESIGN
In this chapter dedicated to system design, we began by examining the key characteristics of our proposed solution. We then provided a detailed description of the overall system pipeline, explaining its main steps. Finally, we offered an in-depth presentation of the proposed solution, detailing the construction of the knowledge graph and the model used for medical diagnosis.

— Chapter 3: SYSTEM IMPLEMENTATION AND TESTING
This chapter presents the implementation of the proposed solution. We describe the various tools used in the development of our system, then move on to explain in detail the complete testing procedure. We analyze the results produced by the explainable AI techniques and provide their interpretation, and finally, we introduce the user interface and the main functionalities of the developed application.

The thesis concludes with a general conclusion that summarizes the key points discussed throughout the work and offers several perspectives for future improvements.

Chapter 1

Knowledge Graph And Explainable AI

1.1 Introduction

In recent years, knowledge graphs (KGs) have become a key area in Artificial Intelligence (AI), driven by the need to make sense of the vast amounts of Big Data. While machines still have difficulty fully understanding natural languages like English, they are better at processing structured data, such as graphs, dictionaries, and tables. This gap highlights the usefulness of knowledge graphs in enabling machines to analyze and reason about data more effectively.

knowledge graph, as a semantic network, has been widely used in natural language processing, intelligent question answering system, intelligent recommendation system and so forth. Together with big data and deep learning, knowledge graph now has become one of the core driving power for the development of artificial intelligence(1).

This chapter aims to present the fundamental concepts on which this research is based, by highlighting the knowledge graph and its increasing importance, we will also review the methods for building knowledge graphs, starting from raw data and progressing to interconnected and inference-capable representations.
Next, we will explore the concept of explainable artificial intelligence, outlining its types and its importance in supporting medical decision-making and enhancing the reliability of systems.
Finally, we will address the related work, where we will present the most recent and relevant research contributions in this field, along with a comparative analysis to identify the strengths and weaknesses of each, and to highlight the research gaps that this work aims to address.

1.2 Knowledge Graph

In this section, we explore the concept of the Knowledge Graph, and illustrative examples of knowledge graphs and how they are used in real-world applications.

1.2.1 Definition of Knowledge Graph

The knowledge graph is a graph-theoretic representation of human knowledge such that it can be ingested with semantics by a machine. In other words, it is a way to express 'knowledge' using graphs, in a way that a machine would be able to conduct reasoning and inference over this graph to answer queries ('questions') in some meaningful way. However, The simplest functional definition of a knowledge graph is that it is a set of triples, with each triple intuitively representing an 'assertion'. If the KG was constructed correctly over a trustworthy data source, we could also think of assertions as facts. Formally, a triple is a 3-tuple (h, r, t) where h represents a head entity, t represents a tail entity, and r expresses a relationship between the two entities. Many, though not all, statements in natural language (e.g., English) can be expressed conveniently in this form. Consider, for example, the sentence Fido the dog stole a bone from Mary's backyard, which can be expressed as a set of triples1 {(Fido, is-a, Dog), (Fido, stole, bone_1), (bone_1, is-a, Bone), (bone_1, located-in, yard_1), (yard_1, is-a, Yard), (yard_1, belongs-to, Mary), (Mary, is-a, Person)}(2)

.

This fascination (both industrial and academic) with graphs aside, there was another good reason to think of knowledge bases as graphs. First, if one takes the step of visualizing the first and third elements (i.e. h and t) of a triple as nodes, and the second element r as a labeled, directed edge pointing from h (the head entity) to t (either a tail entity or an attribute), an intuitive data model emerges (Fig. 1.1). In fact, many people would find it easier to draw the kind of diagram shown in Fig. 1.1 than thinking carefully about sets of triples. In a certain sense, the KG can be said to serve as a lingua franca between machines and humans, in that it is structured enough for machines to process and ingest with semantics, but is intuitive enough for humans to make sense of, at least if represented and drawn using common-sense mnemonics. In fact, the Freebase knowledge graph, and more recently, Wikidata, allow the crowdsourced acquisition of such structured knowledge, as opposed to Wikipedia, where the crowdsourced knowledge is acquired mostly in natural language(2).

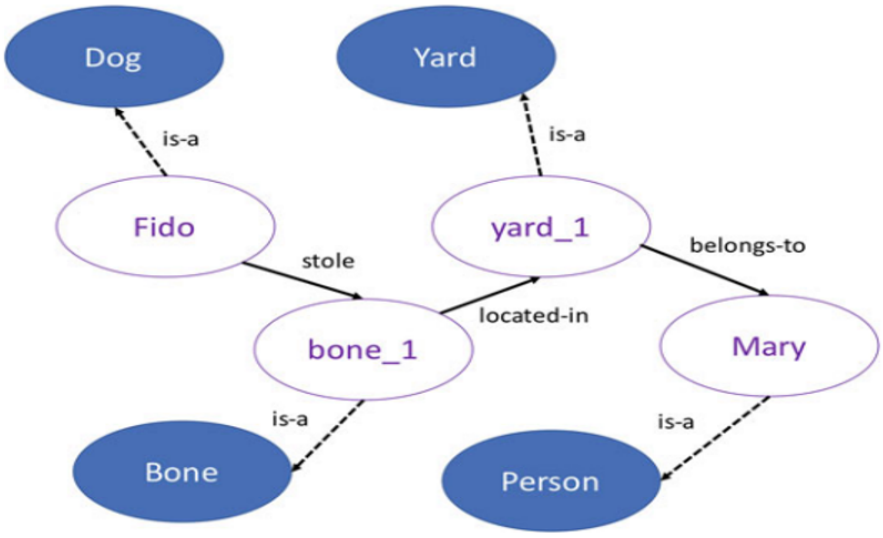

Figure 1.1: The Knowledge Graph representation of the information expressed in the Fido the dog example(2).

Filled ovals (i.e. concepts) are parts of the ontology, while the unfilled ovals are part of the KG itself (the instances) is-a relationships (dashed edges) mediate between instances (in the KG) and concepts (in the ontology). Other relationships are defined in the ontology, but used in the KG.

1.2.2 A Brief History of Knowledge Graphs

Cooking is culture, and culture is based on history. History is not only what has happened, but also what has been piled up the ground upon which we stand and build. Therefore, we should also have an understanding of where knowledge graphs come from if we want to become a maestro KG chef. Understanding the historical context is always paramount to understanding the possible paths one can take in the future.

Fast forward

- In 1736, graph theory was born: Leonhard Euler formulated the 'Königsberg Bridge Problem(4).
- In 1976, John F. Sowa published his first paper on Conceptual Graphs(4).
- In 1982, Knowledge Graphs were invented in the Netherlands. The theory of Knowledge Graphs was initiated by C. Hoede, a mathematician at the University of Twente, and F.N. Stokman, a mathematical sociologist at the University of Groningen(9).
- In 1999, Resource Description Framework (RDF) Model was published as a W3C Recommendation to lay a foundation for a Semantic Web(9).

- In 2012, Google introduced their Knowledge Graph, and since then a lot of companies have started to build their own projects using knowledge graphs in various flavours(9).
- In 2018, The GQL Manifesto (6) was published to agree on a standard for a property graph query language.
- By the end of 2019 knowledge graphs had become mainstream. For example, Gartner states that "... a semantic knowledge graph can be used to power other data management tasks such as data integration in helping automate a lot of redundant and recurring activities(7)."
- After decades of developing KGs, the discipline has also been influenced by a lot of other knowledge domains including mathematical logic, graph theory, information retrieval, computer linguistics, knowledge representation and reasoning, and most recently, the Semantic Web and machine learning(9).

1.2.3 The Importance of Knowledge Graphs and Their Applications

Why should we build a knowledge graph? This section outlines the main motivations for learning how to construct knowledge graphs and demonstrates the benefits they can provide. According to Ehrlinger and Wöß (3), knowledge graphs address many well-known challenges in data and content management. They act as a powerful linking engine for enterprise data systems, enabling automatic generation of unified views from heterogeneous and initially disconnected data sources—such as in Customer 360 use cases. Moreover, KGs produce reusable and enriched datasets that can be leveraged in analytics workflows or for training machine learning models (3).

Knowledge graphs are applied in various domains such as information retrieval, recommendation systems, natural language processing, question-answering systems, healthcare, finance, education, industry, social networks, and e-commerce. They help enrich data, optimize processes, and facilitate the discovery of new knowledge.

- **Knowledge Graph Example**

As the example of a domain-specific KG, let us consider the academic publication domain (Fig. 1.2). The two purple nodes in the center of the KG represent different publications, named mnemonically by their publication titles:

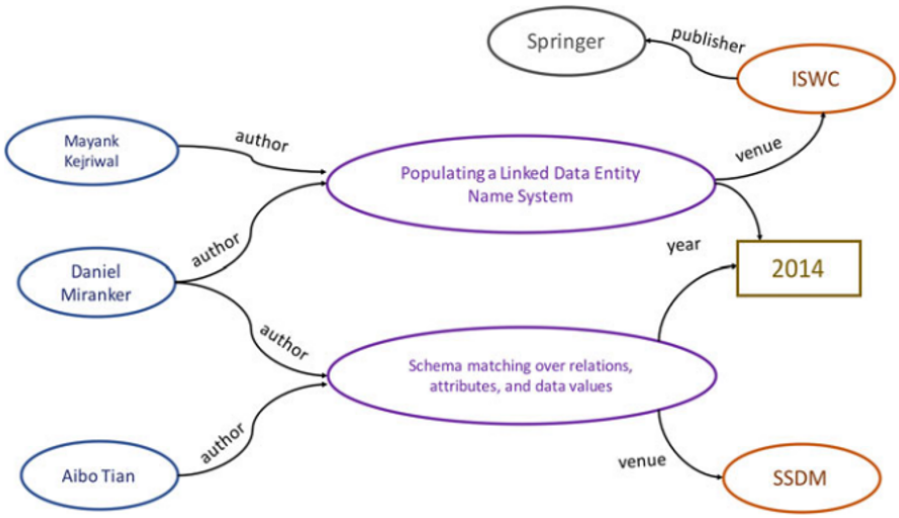

Figure 1.2: An illustration of a Publication knowledge graph, showing two different publications sharing authors. Rectangles are typically used diagrammatically to represent literals while oval nodes represent resources or entities(8).

Some important details concerning the publications are also shown, including their authors, dates of publication and venues.

Despite its simplicity, the KG in Fig. 1.2 illustrates some of the expressiveness in representation, an issue that becomes extremely important in communities such as the Semantic Web. The oval nodes in the figure represent entities or resources, and are generally referred to as Internationalized Resource Identifiers (IRIs), a generalized form of Uniform Resource Identifiers (URIs). Entities have relationships with other entities (such as between authors and their publications) or attributes (such as the year of a publication). The distinction can be expressed by the fact that in a triple (h, r, t), t is either a literal or an entity. Note that h is always an entity(8).

1.2.4 Knowledge Graph Building Phases

A. Knowledge Acquisition:

- **Data Collection:** is the first and foundational step in constructing a Knowledge Graph. It involves gathering raw data from various structured, semi-structured, and unstructured sources. This phase ensures that sufficient, relevant, and high-quality data is available to represent real-world knowledge in a graph format.

- **Entity and Relation Extraction:** is the second critical step in the construction of a Knowledge Graph. It involves identifying and extracting key concepts (entities) and the semantic relationships that link them from raw data sources.

B. Knowledge Hosting:

- This step considers the provenance of the created knowledge from various sources and the contextualization of that knowledge (20).

C. Knowledge Curation:

- This phase aims to assess, correct, and enrich the knowledge graph. It involves evaluating the quality of the knowledge graph in various dimensions and improving it, particularly in terms of correctness and completeness (20).

D. Knowledge Deployment:

- The final phase involves putting the knowledge graph into use, making it accessible for applications and users(20).

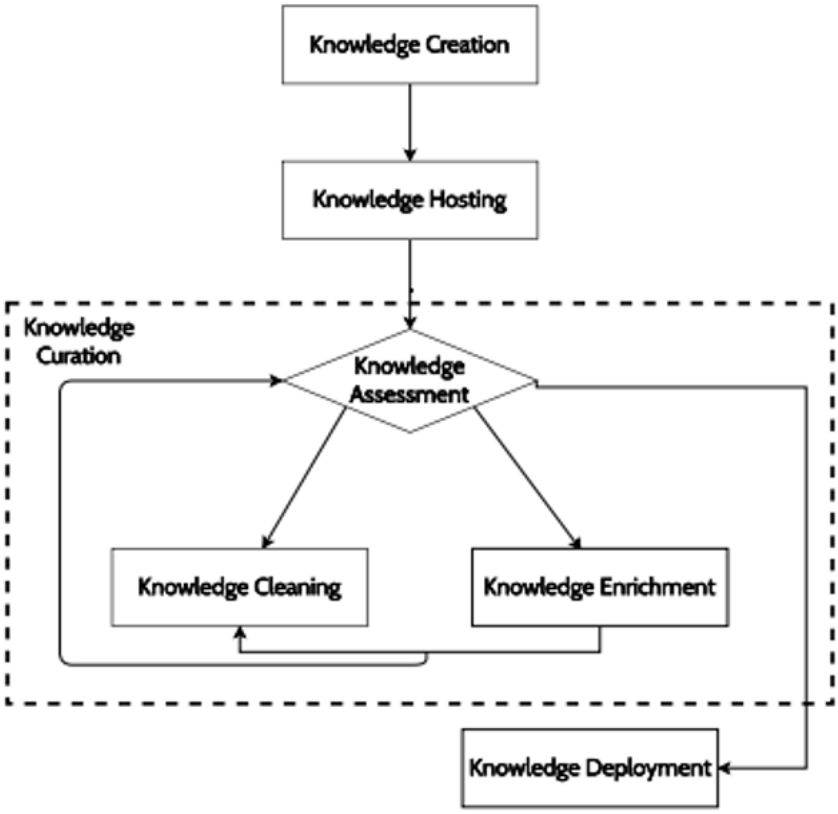

Figure 1.3: A methodology for building knowledge graphs

1.2.5 Tools Used In Knowledge Graph

Graph databases and ontology development tools are essential technologies for managing and reasoning over complex, interconnected data—especially in domains like healthcare,

knowledge graphs, and semantic web applications.

- **Neo4j** is one of the most widely adopted graph databases, known for its property graph model and **Cypher** query language. It enables efficient querying and visualization of deeply connected data, making it suitable for use cases like recommendation systems, fraud detection, and knowledge graph construction. Neo4j supports both native graph storage and processing and provides integration with various programming languages and data science tools(21).

- **Amazon Neptune** is a fully managed graph database service offered by AWS that supports both the property graph model (using Apache TinkerPop/Gremlin) and the RDF model (using SPARQL). It is designed for high performance and scalability in graph-based workloads, with applications in fraud detection, knowledge management, and network security. Neptune integrates well with other AWS services, making it a powerful tool for cloud-based graph applications(22).

-For ontology development, **Protégé** is a free, open-source platform developed by Stanford University for building and managing OWL ontologies. It supports a wide range of reasoning engines and plugins, providing a user-friendly interface for editing complex ontologies and aligning them with semantic web standards. Protégé is extensively used in biomedical informatics and linked data research (23).
These technologies collectively support the creation, querying, and reasoning over rich data structures, enabling the development of intelligent systems across various domains.

1.2.6 Knowledge Graphs and Health Data

We can divide health care data into three categories: omics data, clinical data, and sensor data.
Omics data deal with large-scale and high-dimensional data, which indicates the totality of some kind like genomics, proteomics and transcriptomics (39).
Clinical data contains patients' electronic health records, which are collected as part of ongoing patient care and act as the main resource for most medical and health research (40).
On the other hand, sensor data refers to data collected from numerous wearable and wireless sensor devices such as pulse oximetry, heart rate monitors, blood pressure cuffs, Parkinson's disease monitoring system, and depression-mood monitoring systems (41).
Knowledge graphs constructed based on omic data, more specifically multi-omic data, are used to explore the relationship between such entities and diseases and provide novel discoveries (42). These graphs have been used to identify protein–protein interactions

(42; 43; 44; 45), and gene–disease association (47; 48; 49).
Knowledge graphs can be used to extract novel information from clinical data and improve patient care. Studies have been conducted to recommend safer medicine combinations (50; 51), predict the probability of patient–disease associations like heart failure probability (52) and improve patient diagnoses (51; 53). Knowledge graphs provide the opportunity to convert a variety of healthcare data into a uniform graph format and picture all the links among different objects to create knowledge from diverse fields, like diseases, drugs or treatments, through edges with various labels. This ability of knowledge integration, not available in traditional pharmacologic experiments, can speed up the discovery of knowledge (54).

1.3 Explainable AI

In this section, we discuss XAI, its importance in healthcare, and the different categories of XAI methods.

1.3.1 Definition of XAI

Explainable Artificial Intelligence refers to a suite of techniques and methodologies that make the decisions of AI systems understandable to humans. As machine learning models, particularly deep learning and ensemble models, grow in complexity and predictive power, their internal decision processes become less transparent—often referred to as "black boxes" Explainable AI is gaining popularity (24). XAI aims to transform these opaque systems into ones whose outputs can be explained in human-interpretable terms without significantly compromising performance.

The core objective of XAI is to generate models that can either inherently explain their predictions (ante-hoc interpretability) or provide explanations after predictions have been made (post-hoc interpretability). These explanations may be visual, textual, or based on simplified surrogate models, depending on the use case and audience (25). In essence, XAI enhances the interpretability and trustworthiness of AI, making it possible for users, stakeholders, and regulators to understand why a model made a specific decision.

1.3.2 Importance of XAI in Healthcare

The healthcare sector represents one of the most critical domains for the application of XAI. Medical decisions—such as diagnosis, prognosis, and treatment planning—often carry life-altering consequences, which necessitate not only accuracy but also transparency. Physicians, patients, and healthcare institutions must be able to understand and trust the AI systems that inform clinical decisions (37).

First, explainability fosters trust between clinicians and AI tools. For example, if a model predicts a high likelihood of a certain disease, the doctor need to understand which clinical factors influenced that prediction such as lab results, imaging features, or patient history—before accepting or acting on it (35). Without clear explanations, even highly accurate models risk being rejected by healthcare professionals.

Second, XAI supports accountability and ethical standards. In cases of medical error or misdiagnosis, it's essential to trace the reasoning behind an AI model's decision. Transparent explanations allow for auditing and correction, reducing the risk of systemic bias or malpractice (28).

Lastly, XAI empowers human-in-the-loop systems, where clinicians and AI work collaboratively. Rather than replacing doctors, interpretable AI serves as a decision-support tool, allowing medical professionals to validate or challenge algorithmic predictions using their clinical judgment. This collaboration enhances both efficiency and safety in clinical workflows (29).

In summary, XAI is not just a technical requirement in healthcare—it is a foundational component for ethical, legal, and clinical adoption of AI technologies. Its role in making AI systems transparent, trustworthy, and justifiable is essential for building confidence among practitioners and patients alike.

1.3.3 Categories of XAI Methods

This section breaks down the main categories of XAI methods into two broad approaches:

1. **Post-hoc Approaches:** Post-hoc methods analyze the predictions of an already trained model to provide explanations.

    - **Local Explanations**
        - **LIME (Local Interpretable Model-agnostic Explanations)**

            ➢ **Principle:** LIME generates simple, interpretable models around a specific prediction by perturbing input data. The perturbations help observe the impact of changes in a feature on the prediction of a complex model (30).
            ➢ **Application:** LIME is widely used for complex models such as neural networks.
            ➢ **Advantage:** Provides interpretable local explanations.

➢ **Limitation:** Approximations may lack reliability for certain instances.

– **SHAP (SHapley Additive exPlanations)**

➢ **Principle:** Based on Shapley game theory, SHAP assigns a value to each feature according to its contribution to the prediction. This allows understanding how each input variable influences the model's output(31).

➢ **Application:** Often used in healthcare applications to explain why a prediction was made (e.g., disease risk).

➢ **Advantage:** Provides precise and fair interpretations of features.

➢ **Limitation:** Resource-intensive for complex models.

- **Global Explanations**
  - **SHAP (SHapley Additive exPlanations - Global)**

➢ **Principle:** SHAP values can be aggregated to provide an overview of feature importance across the entire model(31).

➢ **Application:** Useful for healthcare professionals to identify the most influential features across all predictions.

➢ **Advantage:** Offers a global understanding of the model.

➢ **Limitation:** Requires intensive computations for large datasets.

  - **Partial Dependence Plots (PDP)**

➢ **Principle:** PDP analyzes the impact of a specific feature on the prediction while keeping other features constant. This visualizes the relationship between a variable and the model's output(32).

➢ **Application:** In the medical field, PDP can show how age or glucose levels affect the predicted risk of a disease.

➢ **Advantage:** Simple to interpret and visually intuitive.

➢ **Limitation:** Does not capture complex interactions between variables.

2. **Ante-hoc Approaches:** Ante-hoc methods are inherently explainable because they are designed to be interpretable from the start.
   - **Decision Trees**

➢ **Principle:** Decision trees use a node-based structure to make decisions. Each node represents a condition on a variable, and the leaves represent the final outcomes.

➢ **Application:** Suitable for medical applications where transparency is crucial(33).

➢ **Advantage:** Easy to interpret for non-technical users.

➢ **Limitation:** Performs poorly on complex relationships or non-linear data.

– **Linear Models (e.g., Logistic Regression)**

➢ **Principle:** Linear models use coefficients to indicate the impact and direction of features on the predicted outcome(34).

➢ **Application:** Often used to predict probabilities in medical diagnoses.

➢ **Advantage:** Interpretable and fast to compute.

➢ **Limitation:** Does not perform well with complex non-linear relationships.

1.3.4 Applications of XAI In Healthcare

The healthcare sector is one of the most promising and critical fields for the application of XAI. As AI models are increasingly used for tasks such as disease diagnosis, risk prediction, and treatment planning, the demand for transparency and trust becomes vital. Physicians and medical professionals are often hesitant to adopt black-box AI systems unless the decision-making process is understandable and verifiable (35).

Post-hoc methods like SHAP have proven particularly useful in explaining predictions made by deep learning or ensemble models in clinical contexts. For example, SHAP can clarify which factors (e.g., age, blood pressure, glucose levels) most influenced a model's decision to classify a patient as high risk for diabetes or cardiovascular disease (36). LIME has also been applied to interpret imaging-based diagnostic models, offering localized explanations that highlight which regions of an X-ray or MRI were most influential in determining the diagnosis (37).

In addition, ante-hoc models such as decision trees are still widely used in clinical decision support systems, especially in early screening tools and triage applications, due to their simplicity and interpretability. Expert systems and rule-based models are also employed in settings where clinical guidelines can be codified into structured IF-THEN rules, such as diagnosing common infections or managing chronic diseases (38).

The integration of explainability not only improves user confidence but also supports better patient outcomes by enabling shared decision-making between AI systems and healthcare

providers.

1.4 Comparison of Related Works

In this section we present some of the related works works that have contributed to the field of medical diagnosis:

1. (Yu, G et al.2022)(10): This article is entitled "Improving the management of chronic diseases for children with knowledge and artificial intelligence charts." This research explores innovative approaches to promoting health management for children facing chronic diseases, using advanced technologies such as artificial intelligence and big data. The authors, including experts from different institutions, aim to address the challenges in managing chronic diseases and propose effective solutions to improve healthcare outcomes for young patients, highlighting the contribution of knowledge graph and artificial intelligence in this context. The research presented in the article uses many methodologies and techniques to develop a chronic management system for artificial intelligence (AICMS) to manage chronic diseases in children.

2. (Najjar, 2023) (11): This article is entitled "Redefining radiation: reviewing the integration of artificial intelligence into medical imaging" by Ripal Najjar. This article explores the transformative role of artificial intelligence in radiology, explaining how deep learning, especially CNNs, revolutionized image analysis tasks within health care, highlighting the technology's contribution to improved classification, fragmentation, computer-aided detection, predictive analysis and improved workflow, revealing the impact of artificial intelligence on enhanced diagnosis and improved patient quality of care.

3. (Chai, X. 2020)(12): This article titled "Diagnosis Method of Thyroid Disease Combining Knowledge Graph and Deep Learning". The main objective of the study is to propose a diagnosis method for thyroid diseases that combines knowledge graphs and deep learning techniques. This approach aims to enhance the accuracy and efficiency of disease diagnosis by leveraging structured medical knowledge and advanced computational methods. The study seeks to address the limitations of traditional diagnostic systems by utilizing a comprehensive knowledge graph that incorporates various entities and their relationships, ultimately providing intelligent diagnostic support for both patients and healthcare professionals.

4. (Mazo et al., 2021) (13): The work presented in the document focuses on the integration of Explainable Artificial Intelligence within Clinical Decision Support Systems (CDSS) to enhance the interpretability and trustworthiness of machine learning models

in healthcare. It explores the challenges associated with unstructured medical data and emphasizes the importance of using interpretable algorithms to facilitate clinician understanding and rationalization of model predictions. The study highlights various AI techniques employed in CDSS, specific diagnostic tasks addressed, and the performance metrics used to evaluate these systems. Additionally, it discusses the constraints and limitations of current approaches, advocating for interdisciplinary collaboration to improve the implementation and acceptance of AI solutions in clinical practice. Overall, the work aims to bridge the gap between AI outputs and human explanations, ultimately promoting safer and more reliable healthcare decision-making.

5. (Kim et al., 2024) (14): This work presents a novel framework for Explainable Artificial Intelligence based Clinical Decision Support Systems (CDSSs) aimed at enhancing transparency and interpretability in medical decision-making. The framework integrates various AI techniques, including scoop-based, model-based, complexity-based, and methodology-based approaches, to address the black-box problem commonly associated with traditional AI models. By utilizing multimodal data—such as text, audio, images, and genomic information—the proposed system supports a range of diagnostic tasks, including monitoring, diagnosis, and prescription. The authors systematically review existing CDSSs, highlighting their limitations in data utilization and interpretability, and propose a forward-looking roadmap for implementing XAI technologies in healthcare. This framework not only aims to improve the accuracy and reliability of AI-driven decisions but also ensures that the rationale behind these decisions is clearly communicated to users, thereby fostering trust and effective management in clinical settings.

6. (Xie et al., 2024) (15): This paper introduces GRAPHCARE, a framework that improves healthcare predictions by utilizing personalized knowledge graphs derived from external medical knowledge and large language models. Evaluated on the MIMIC-III and MIMIC-IV datasets, GRAPHCARE demonstrates superior performance in key tasks such as mortality prediction, readmission prediction, length of stay estimation, and drug recommendation. The framework employs a Bi-attention AugmenTed graph neural network, achieving significant accuracy improvements while requiring fewer patient records. The study highlights the potential of integrating external knowledge into healthcare predictions, promoting personalized medicine and enhancing patient care.

1.4.1 Comparison Of Related Works:

According to the presented work, we conducted a comparison using a table 1.1, that contains the following properties :

- **Dataset:** The specific datasets used for training and evaluation of the models, which

can impact the generalizability of the results.

- **AI Techniques Used:** The methodologies and technologies employed in the model, such as knowledge graphs, machine learning algorithms, or natural language processing techniques.
- **Specific Diagnostic Tasks:** The particular tasks the model is designed to perform, such as diagnosis prediction, summarization, or concept extraction.
- **Performance:** Metrics that indicate how well the model performs, including accuracy (correct predictions), F1-score, precision (correct positive predictions).
- **Constraints or Limitations:** Challenges faced by the model, such as scalability (ability to handle larger datasets), computational complexity (resources required for processing), and data quality issues.

Article	Dataset	Techniques Used	Specific Diagnostic Task	Performanc	Constraints or Limitations
Yu, G et al., 2022	Electronic health records, patient monitoring data, and patient-reported information	Knowledge graphs, machine learning (logistic regression, naive Bayes)	Recommending hospital visits, prediction of asthma attacks, monitoring chronic conditions	AUC = 0.83 for regression model; Not mentioned for Naive Bayes	Scalability issues with large datasets; real-time computational complexity; need for continuous updates to maintain data quality
Najjar, 2023	Medical images (X-ray, CT, MRI)	Convolutional Neural Networks (CNNs)	Disease classification (tumor classification in mammography); image segmentation (lung nodules from CT scan)	Accuracy = 87%; Precision = 93%	High computational complexity; large data requirements; no interpretability (black box problem)

Article	Dataset	Techniques Used	Specific Diagnostic Task	Performanc	Constraints or Limitations
Chai, X., 2020	Hospital database (patient info, diagnosis data, drug use information)	Knowledge graphs + deep learning (CNN, SVM, LSTM)	Diagnosis of thyroid diseases, relationship extraction and inference	Accuracy = 91.36% (KG+CNN); Accuracy = 92.23%, F1 = 91.34% (KG+LSTM); Accuracy = 88.11% (KG+SVM)	Computational complexity due to deep learning; scalability issues with large datasets; NO interpretability.
Mazo et al., 2021	Various electronic health record datasets	Supervised/Deep Learning, Ensemble Methods, NLP, Post-hoc and Ante-hoc XAI, Model-diagnostic Techniques	Diagnosis support, treatment recommendations, prognostic assessments, follow-up care decisions	95% accuracy for cardiac pathology classification (Zheng et al.)	Unstructured data challenges; Potential systemic bias in ML models; lack of XAI evaluation in clinical settings; need for interdisciplinary collaboration and user-centered approaches to bridge gaps in explainability
Kim et al., 2024	Multimodal clinical datasets	Explainable AI technologies, deep learning models	Monitoring, diagnosis, prescription, warnings, document management	Not mentioned	Limited data utilization range; opacity in decision-making

Article	Dataset	Techniques Used	Specific Diagnostic Task	Performanc	Constraints or Limitations
Jiang, P., Xiao, C., Cross, A., & Sun, J., 2024	MIMIC-III and MIMIC-IV datasets	Bi-attention Augmented Graph Neural Network (BAT GNN), Large Language Models (LLMs)	Mortality prediction, readmission prediction, length of stay, drug recommendation	MIMIC-III: 43.1%; MIMIC-IV: 45.7% (for GRAPH-CARE with GAT)	Dependent on dataset quality; requires expert validation of knowledge graph triples

Table 1.1: Comparison of Related Works

1.4.2 Discussion Of Related Works

In this section, we analyze related works based on key criteria, including the type of dataset used, the techniques implemented, specific diagnostic tasks, performance metrics, and identified limitations.

1. Dataset

The related works can be classified based on dataset type: structured vs. unstructured and public vs. Proprietary

- **Structured vs. Unstructured:** Yu, Chai, Mazo, and Jiang leverage structured EHR or clinical-database records, which facilitate tabular modeling and knowledge-graph construction. Najjar and Kim include rich unstructured data (images, multimodal streams), unlocking vision and signal processing approaches but requiring heavy annotation pipelines.
- **Public vs. Proprietary:** Only Jiang and Mazo use publicly accessible MIMIC or multi-center EHR datasets, enhancing reproducibility. The others rely on proprietary hospital records or mixed sources, which can impede external validation.

2. Specific Diagnostic Task

The related works can also be categorized by diagnostic task type: predictive vs. Classification

- **Predictive vs. Classification:** Yu and Jiang prioritize time-to-event and risk predictions (asthma attack, mortality, readmission), valuable for proactive care. Najjar, Chai, and Mazo span classification (tumors, thyroid, diagnostic support) and recommendation tasks, illustrating a broad utility of AI in diagnostics and treatment planning.

3. Techniques Used

The related works can also be categorized by technique type:

- **Classical ML vs. Deep Learning:** Yu and Chai blend interpretable classical ML (logistic regression, SVM) with graph or DL components, trading off some performance for clarity. Najjar and Kim rely heavily on deep convolutional nets and explainable DL frameworks to tackle image or multimodal tasks.

- **Explainability Emphasis:** Mazo leads in integrating both ante-hoc (transparent model architectures) and post-hoc (saliency, SHAP, LIME) XAI methods. Jiang's BAT-GNN + LLM pipeline shows a novel graph-based, attention-augmented interpretability layer.

- **Graph-Based Transparency:** Knowledge-graph methods (Yu, Chai, Jiang) inherently represent relationships in an intelligible node-edge format, boosting clinician trust. Mazo and Kim push further with integrated explanations—highlighting WHY a model decided as it did, rather than simply WHAT the output is.

- **Black-Box Concerns:** CNN-centric Najjar and some deep DL components in Kim retain "black-box" opacity, underscoring an ongoing trade-off between raw performance and explainability.

4. Performance

Mazo et al. (2021) reported the highest accuracy at 95%, showing strong results for cardiac pathology classification. Chai (2020) closely follows with up to 92.23% accuracy and 91.34% F1-score using a combination of knowledge graphs and deep learning. Najjar (2023) achieved 87% accuracy and 93% precision in medical image tasks, reflecting the challenge of image-based diagnostics. Yu et al. (2022) reached an AUC of 0.83, which is solid for real-time prediction from diverse health data. While Jiang et al. (2024) reported 43–45%, the lower score reflects the complexity of multimodal hospital data rather than poor model performance.

5. Constraints or Limitations

- **Compute Scalability:** Nearly every study flags high computational overhead—especially Najjar's image pipelines and DL-heavy Chai and Kim architectures—plus the need for continuous data updates (Yu).

- **Data Quality Bias:** Mazo and Jiang explicitly call out unstructured-data challenges and potential KG biases from LLMs. The reliance on single-center or proprietary datasets raises concerns about generalizability across diverse patient populations.

- **Clinical Validation:** Only Mazo et al. mention the shortage of real-world XAI trials. The others implicitly require further prospective studies and expert-in-the-loop validation before deployment.

1.5 Conclusion

In this chapter, we explored the fundamental concepts underpinning this work, namely knowledge Graphs and Explainable Artificial Intelligence. We provided a comprehensive overview of the key theoretical foundations related to these concepts. We explained how medical knowledge can be represented using knowledge graphs and discussed the mechanisms involved in constructing them to support a better understanding of the relationships between medical entities. Additionally, we examined the concept of explainable AI and its various types.

The chapter concluded with a review of rlated works, through which we analyzed several research works related to our topic. This analysis enabled us to identify existing gaps in the field and guided the direction of our project in a way that aims to address these gaps.

Chapter 2

System Design

2.1 Introduction

Design is a critical phase in the development of any system. In this chapter, we present the design stages of our system, which aims to interpret the decisions of deep learning models based on knowledge graphs. We begin by introducing the key components of the system, followed by an overview of our proposed solution and the overall system architecture. Then, we detail the implementation steps, starting with the construction of the knowledge graph—achieved by selecting and processing a suitable dataset to generate a complete graph. We proceed to extract the knowledge graph embeddings to be used as inputs to our model. Subsequently, we describe the development of a knowledge graph-based model for classification tasks, concluding with the interpretation of the model's decisions using state-of-the-art explainable AI techniques.

2.2 Features of our System

As part of our project, we developed a solution in the form of a medical diagnosis system that combines knowledge Graphs and Explainable AI. This system is characterized by several key features, most notably:

- **Dataset Used:**

  We selected a dataset named Lung Cancer Dataset, which focuses on lung cancer due to its widespread prevalence and high sensitivity in the medical field—making its study and data analysis particularly critical. This dataset is of high quality, containing no null values, and is available in multiple formats such as CSV and JSON, which facilitates its integration and manipulation within the system.

- **Techniques Used:**
  The developed system relies on a set of advanced techniques:

- Knowledge Graphs: Used to represent medical knowledge through interconnected concepts such as symptoms, diseases, and influencing factors.

- Graph Attention Networks: GATs are a type of deep learning model from the Graph Neural Network (GNN) family, designed to represent and process structured data in the form of graphs. They leverage an Attention Mechanism to assign relative importance to each connected node, thereby improving prediction accuracy.

- **Specific Diagnosed Task:**
  The primary diagnostic task is to classify a patient's medical diagnosis as either affected or unaffected by lung cancer. The system also provides justifications for each diagnostic decision, highlighting the underlying factors involved.

- **Explainability:**
  One of the most critical features of the proposed system is its explainability, which is essential in sensitive medical applications. The system integrates two complementary approaches:

  - Local Explainability: Shows the impact of each input feature (e.g., symptoms, clinical indicators) on a specific patient's prediction.

  - Global Explainability: Quantifies the contribution of each feature to the final prediction, providing a detailed view of how decisions are made.

Compared to existing approaches, the proposed system offers a more balanced integration of accuracy, explainability, and reliability—factors that rarely exist in current models. Most existing models tend to focus on achieving high predictive accuracy while neglecting interpretability and understanding of model behavior.

We specifically chose Graph Attention Networks due to their strong ability to capture complex relationships in medical data and their proven effectiveness in clinical prediction tasks, as reported in prior studies (e.g., Jiang et al., 2024). However, given the inherently "black-box" nature of deep learning models, we integrated Explainable AI techniques to enhance interpretability, build trust among medical professionals, and ensure the verifiability and understandability of the system's decisions. This directly addresses critical limitations highlighted in recent research (e.g., Kim et al., 2024; Mazo et al., 2021).

2.3 Schema of our system

The primary objective of this project is to develop a medical diagnosis system based on the integration of Knowledge Graphs with advanced artificial intelligence techniques.

Additionally, we apply Explainable Artificial Intelligence techniques to provide clear interpretations of the diagnostic results.
To simplify and clarify the overall process, we designed a comprehensive diagram illustrated in Figure 2.1, which summarizes the main steps of our project.

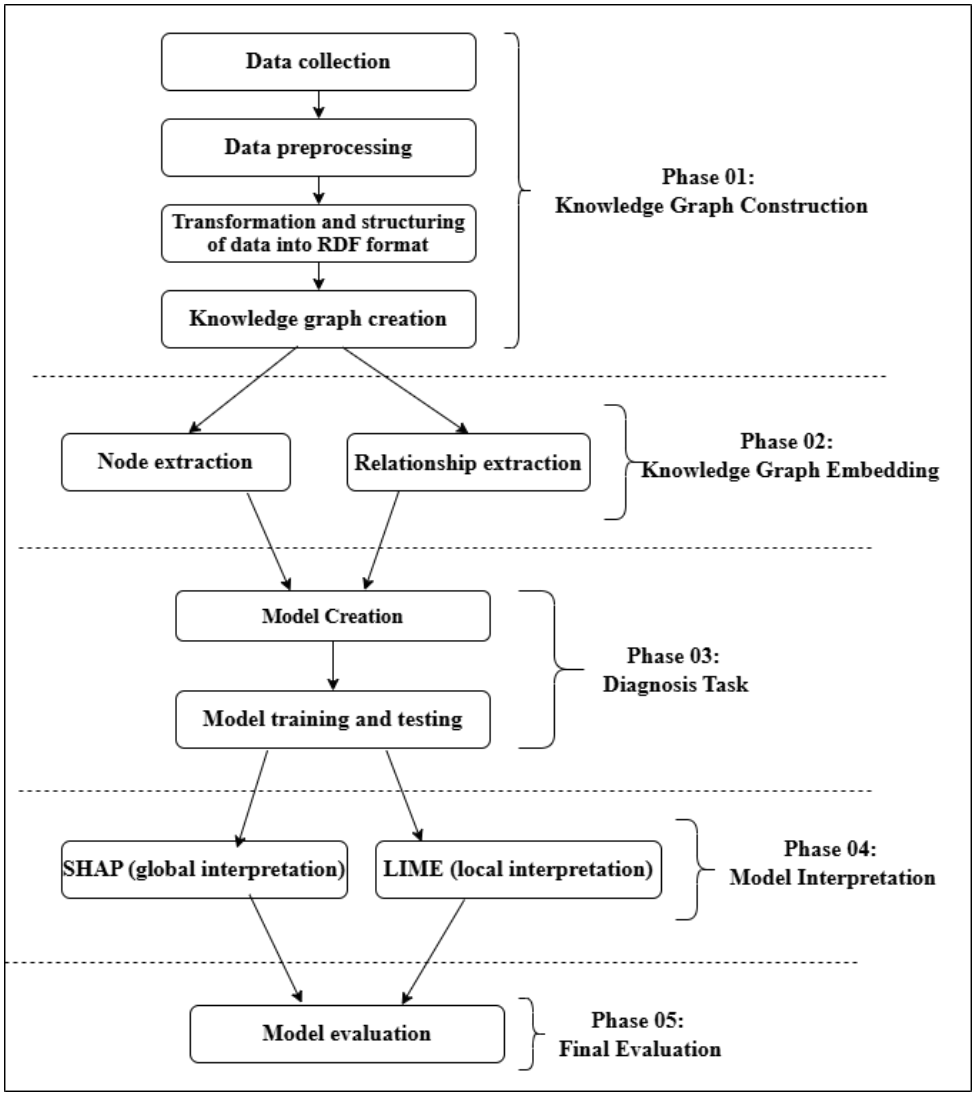

Figure 2.1: Global schema of system

Our work consists of five main phases, each comprising several interconnected steps:

A. Phase 1: Knowledge Graph Construction

We begin by selecting an appropriate dataset, followed by a cleaning process that involves removing duplicate entries and correcting categorical values. Subsequently, we define the RDF graph structure and extract the essential entities and relationships, which are then used to construct the knowledge graph, so we will have four steps in this phase:

- Step 1: Data collection.
- Step 2: Data preprocessing.

- Step 3: Transformation and structuring of data into RDF format.
- Step 4: Knowledge graph creation.

B. Phase 2: Knowledge Graph Embedding
After building the knowledge graph, we proceed to generate node embeddings by selecting an effective algorithm. These embeddings will later serve as inputs to a deep learning model. In this phase we will have two steps:

- Step 1: Node Extraction.
- Step 2: Relationship Extraction.

C. Phase 3: Prediction Task
After extracting the knowledge graph embeddings, they are used as inputs to deep learning models. In this process, we create and experimented with two different models and evaluated their performance using the Accuracy and F1-score metrics. This stage consists of two main steps to ensure optimal results:

- Step 1: Model Creation.
- Step 2: Model training and testing.

D. Phase 4: Model Interpretation
This is considered the most critical phase of the project, where the model's predictions are interpreted. We focus on providing **post-hoc** explanations (i.e., after the prediction) by applying the latest Explainable AI techniques, specifically **LIME** for local explanations and **SHAP** for global explanations.

E. Phase 4: Model Evaluation
In this phase, we assess the quality of the knowledge graph using specific evaluation metrics, in addition to testing the model.

2.4 Description of the proposed solution

In this section, we provide a comprehensive and detailed description of our proposed solution, structured phase by phase to highlight the role of each component in the overall system.

2.4.1 Phase 01: Knowledge Graph Construction

The first phase focuses on building a robust knowledge graph to represent medical information in a structured and unified manner. This phase involves following steps:

2.4.1.1 Data collection

Lung cancer is one of the most common and dangerous types of cancer in the world. According to the World Health Organization, lung cancer remains the leading cause of cancer-related deaths, accounting for approximately 1.8 million deaths in 2020 alone(55). Despite advances in medical research, early detection and effective treatment continue to pose significant challenges, and as part of efforts to understand and control this disease, this project relied on a comprehensive dataset available on the Kaggle platform, which was downloaded specifically for the purpose of analysis and study and can be accessed via the following link: https://www.kaggle.com/datasets/akashnath29/lung-cancer-dataset. (56)

This dataset includes various data related to several factors that influence the risk of lung cancer, including demographic information, lifestyle habits, and common symptoms associated with this disease. These data can be used to build predictive models and identify potential factors leading to infection, contributing to improved prevention, diagnosis and treatment strategies.
The dataset adopted in this project consists of 3,000 observations distributed over 16 variables. Each observation represents a specific individual, while each variable expresses a potential characteristic or symptom related to lung cancer. We present in Table 2.1 the description of the columns of our dataset:

Figure 1.1 illustrates a part of dataset before the data processing:

GENDER	AGE	SMOKING	YELLOW_FINGERS	ANXIETY	PEER_PRESSURE	CHRONIC_DISEASE	FATIGUE	ALLERGY	WHEEZING	ALCOHOL_CONSUMING	COUGHING	SHORTNESS_OF_BREATH	SWALLOWING_DIFFICULTY	CHEST_PAIN	LUNG_CANCER
M	65	1	1	1	2	2	1	2	2	2	2	2	2	1	NO
F	55	1	2	2	1	1	2	2	2	1	1	1	2	2	NO
F	78	2	2	1	1	1	2	1	2	1	1	2	1	1	YES
M	60	2	1	1	1	2	1	2	1	1	2	1	2	2	YES
F	80	1	1	2	1	1	2	1	2	1	1	1	1	2	NO
F	58	1	1	1	2	2	2	2	1	2	2	1	1	2	YES
F	70	1	1	1	2	2	1	2	2	2	2	2	2	1	YES
F	74	2	2	1	1	1	1	2	1	1	1	1	2	1	NO
M	77	1	2	1	2	1	1	1	1	2	1	1	1	2	NO

Figure 2.2: Part of dataset before cleaning.

Colonne	Type	Description
GENDER	Categorical (M–male, F–female)	Biological sex of the patient.
AGE	Numerical	Age of the patient in years.
SMOKING	Ordinal (1–No, 2–Yes).	Smoking behavior.
YELLOW_FINGERS	Ordinal (1–No, 2–Yes)	Yellow-stained fingers, typically caused by nicotine.
ANXIETY	Ordinal (1–No, 2–Yes)	Psychological anxiety in the patient.
PEER_PRESSURE	Ordinal (1–No, 2–Yes)	Influence from peers related to lifestyle decisions (e.g., smoking).
CHRONIC_DISEASE	Ordinal (1–No, 2–Yes)	Presence of any chronic condition (not necessarily lung cancer).
FATIGUE	Ordinal (1–No, 2–Yes)	Experience of persistent tiredness.
ALLERGY	Ordinal (1–No, 2–Yes)	Whether the patient has allergies.
WHEEZING	Ordinal (1–No, 2–Yes)	Presence of wheezing in the patient's breathing.
ALCOHOL_CONSUMI	Ordinal (1–No, 2–Yes)	Alcohol consumption habits.
COUGHING	Ordinal (1–No, 2–Yes)	Occurrence of chronic coughing.
SHORTNESS_OF_BR	Ordinal (1–No, 2–Yes)	Difficulty breathing (dyspnea).
SWALLOWING_DIFF	Ordinale Ordinal (1–No, 2–Yes)	Trouble swallowing.
CHEST_PAIN	Ordinal (1–No, 2–Yes)	Reports of pain in the chest area.
LUNG_CANCER	Categorical (YES, NO)	Diagnostic label indicating whether the patient has lung cancer (target variable).

Table 2.1: Dataset Description Table

2.4.1.2 Data preprocessing

Data processing is an essential step in any project based on data analysis, as it forms the basis on which the accuracy of results and the effectiveness of predictive models are built. In this project, the processing phase included a set of important procedures that contributed to improving the quality of the lung cancer database, allowing it to be effectively exploited in subsequent analysis stages.
The data processing steps included the following:

- **Exclusion of variables not affecting the analysis:** In the context of improving data quality and simplifying its structure, only one column, **GENDER**, was deleted, after ensuring that it did not affect the results of the analysis or the accuracy of the predictive model adopted in this project, which made it unnecessary in the context of the current study. This conclusion was further validated by a domain expert, confirming the irrelevance of this variable to the study's objectives.

We examined the impact of the variable **GENDER** on the target variable **LUNG_CANCER**. To assess the significance of this variable, the **Chi-Square Test of Independence** was applied, which is a widely recognized statistical criterion used to measure the existence of a significant relationship **between two categorical variables**.
The analysis results showed the following:
The p-value obtained from the test was **0.398**, which is considerably higher than the commonly accepted significance level of **0.05**.
This means that we fail to reject the **null hypothesis**, indicating that there is no statistically significant relationship between gender and the likelihood of developing lung cancer in this sample.

- **Deleting duplicate records (Repeated lines):** Among the basic steps in data processing is the process of deleting duplicate records, which aims to improve data quality and ensure the accuracy of the results. During this stage, the database was scanned to detect any duplication in the records, where some repeated rows are found. These duplicates have been omitted to maintain the validity of the analysis. The presence of duplicate observations can distort the results of predictive models and give unequal weight to some cases, reducing the credibility of the results drawn from the analysis. The following algorithm illustrate this treatment:

Pseudo-algorithm 1: *Removing Duplicate Rows from the Database*

Input : D : database containing duplicates
Output : D_{clean} : database without duplicate rows

START

$D_{\text{clean}} \leftarrow$ EmptyDatabase()
foreach *row* $\in D$ **do**
    if *row* $\notin D_{clean}$ **then**
        AddRow(D_{clean}, row)
    end
end
return D_{clean}
END

- **Converting variables into numerical form (Encodage):** As part of the data preparation process for programming, a process of systematic conversion of qualitative variables into digital formats was implemented. This process included modifying symbolic and textual values to become subject to computer processing without affecting their analytical meaning.

For the LUNG_CANCER column, which represents the target variable in this project, the text values "YES" and "NO" were converted to numeric values 1 and 0, respectively, clearly reflecting the presence or absence of the disease. As for the rest of the columns with binary values that were represented by the numbers 1 and 2, they were converted to 0 and 1, respectively, with the aim of unifying the data format and facilitating its processing using classification models.

This type of coding contributes to enhancing the software exploitability of data, and also ensures its compatibility with the requirements of predictive models that rely on digital inputs to achieve more accurate and reliable performance. We present this treatment in this algorithm:

Pseudo-algorithm 2: *Encoding categorical variables in numeric format*

Input : D : database containing categorical variables
Output : D_{encoded} : database with variables transformed into numeric format

```
START
D_encoded ← D
foreach row ∈ D_encoded do
    if row["LUNG_CANCER"] == "YES" then
        row["LUNG_CANCER"] ← 1
    end
    else if row["LUNG_CANCER"] == "NO" then
        row["LUNG_CANCER"] ← 0
    end
    foreach attribute ∈ BinaryAttributes do
        if row[attribute] == 1 then
            row[attribute] ← 0
        end
        else if row[attribute] == 2 then
            row[attribute] ← 1
        end
    end
end
return D_encoded
END
```

The following table gives an example of converting the values of some columns:

Figure 2.2 illustrates a part of dataset after the data processing:

Columns	Original value	Value after conversion	Notes
LUNG_CANCER	YES	1	The presence of lung cancer
LUNG_CANCER	NO	0	No lung cancer
SMOKING	1	0	non-smoker
SMOKING	2	1	smoker

Table 2.2: Encoding process illustrative table.

Id_patient	AGE	SMOKING	YELLOW_FINGERS	ANXIETY	PEER_PRESSURE	CHRONIC_DISEASE	FATIGUE	ALLERGY	WHEEZING	ALCOHOL_CONSUMING	COUGHING	SHORTNESS_OF_BREATH	SWALLOWING_DIFFICULTY	CHEST_PAIN	LUNG_CANCER
patient_1	65	0	0	0	1	1	0	1	1	1	1	1	1	0	0
patient_2	55	0	1	1	0	0	1	1	1	0	0	0	1	1	0
patient_3	78	1	1	0	0	0	1	0	1	0	0	1	0	0	1
patient_4	60	1	0	0	0	1	0	1	0	0	1	0	1	1	1
patient_5	80	0	0	1	0	0	1	0	1	0	0	0	0	1	0
patient_6	58	0	0	0	1	1	1	1	0	1	1	0	0	1	1
patient_7	70	0	0	0	1	1	0	1	1	1	1	1	1	0	1
patient_8	74	1	1	0	0	0	0	1	0	0	0	0	1	0	0
patient_9	77	0	1	0	1	0	0	0	0	1	0	0	0	1	0
patient_10	67	1	1	1	1	0	1	1	0	1	0	1	0	0	0

Figure 2.3: Part of dataset after cleaning.

2.4.1.3 Transformation and structuring of data into RDF format

This stage aims to define the conceptual structure of the knowledge graph. After collecting and preparing the data, the next crucial step is to transform and structure it using an RDF Schema. The RDF Schema provides a flexible and standardized way to represent information and the relationships between different entities within a knowledge graph.

RDF Schema is one of the fundamental components that enables the organization and semantic description of data in a logical manner. RDFS is an extension of the RDF (Resource Description Framework) language and is used to define classes, properties, and the relationships between them. It provides a structural framework that organizes concepts in a hierarchical and logical way within the knowledge graph.

The importance of RDFS lies in its ability to define the hierarchical structure of data, allowing for consistent and well-organized information modeling. This, in turn, enhances the quality and accuracy of the knowledge graph in representing real-world domains.

We present our RDF Schema that construction of schema is done by the help of expert, in the form of a simple and organized pseudo-algorithm without getting into the complexities of the formal RDF syntax:

This algorithm defines four fundamental classes, each representing a key concept in the knowledge model:

- **Person**: Represents an individual or patient.
- **Symptom**: Represents a symptom associated with a disease.
- **RiskFactor**: Represents a risk factor that could contribute to developing a disease.
- **Disease**: Represents a specific disease, with a focus on lung cancer.

Also, a set of properties that link different entities or describe patient properties:

- **age**: A property that specifies a person's age (of type xsd:integer).
- **hasSymptom**: A property that links a person to a medical symptom they are experiencing.
- **hasFactor**: A property that links a person to an associated risk factor.
- **diagnosedWith**: A property that links a person to the disease they have been diagnosed with.

Finally, the characteristics of each patient were defined into categories with the help of an expert. as follows:

- **Symptoms are:** yellowFingers, anxiety, chronicDisease, fatigue, allergy, wheezing, coughing, shortnessOfBreath, swallowingDifficulty, chestPain.
- **Risk Factors are :** smoking, peerPressure, alcoholConsuming.
- **Disease is :** lungCancer.

Algorithm 3 RDF Schema of Lung Cancer dataset

Initialisation:

30 Define Prefixes:
rdf ← "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
rdfs ← "http://www.w3.org/2000/01/rdf-schema#"
xsd ← "http://www.w3.org/2001/XMLSchema#"
lung ← "http://example.org/lungcancer/"

31 Define Classes:
`Class Person`,
`Class Symptom`,
`Class RiskFactor`,
`Class Disease`

32 Define Properties:
`Property age`: Domain: `Person`, Range: `xsd:integer`
`Property hasSymptom`: Domain: `Person`, Range: `Symptom`
`Property hasFactor`: Domain: `Person`, Range: `RiskFactor`
`Property diagnosedWith`: Domain: `Person`, Range: `Disease`

33 Define subClassOf **`Symptom`**: `yellowFingers`, `anxiety`, `chronicDisease`, `fatigue`, `allergy`, `wheezing`, `coughing`, `shortnessOfBreath`, `swallowingDifficulty`, `chestPain`

34 Define subClassOf **`RiskFactor`**: `smoking`, `peerPressure`, `alcoholConsuming`

35 Define subClassOf **`Disease`**: `lungCancer`

36 **END**

Now we present RDF Schema in figure 2.4:

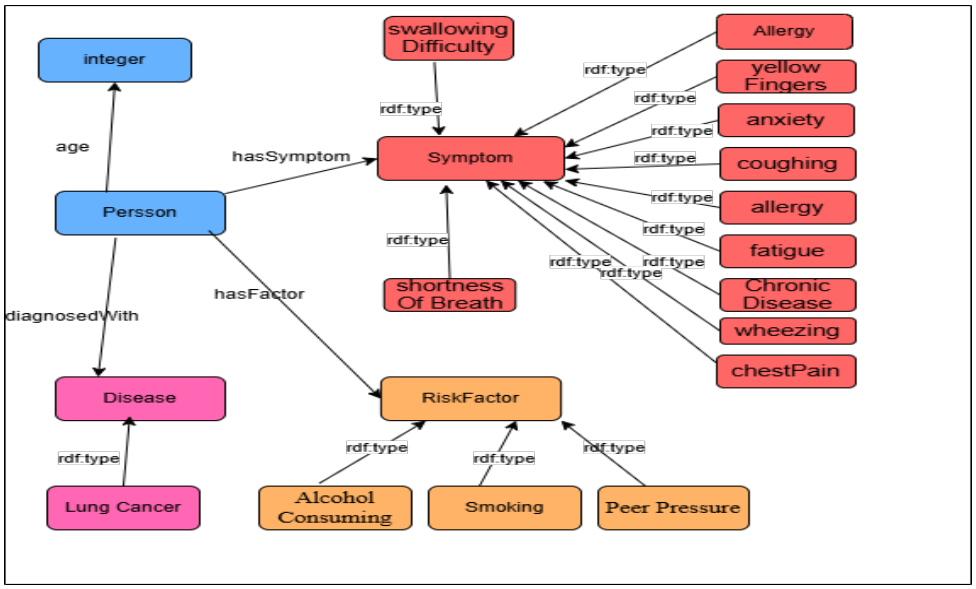

Figure 2.4: RDF Schema Representation of Lung Cancer Diagnosis Ontology

2.4.1.4 Knowledge graph creation

This stage is considered one of the most crucial phases of the project, In this phase, we constructed the knowledge graph, relying on the entities (classes) and ralationships that extracted from the RDF Schema, with a set of rules that contributed to defining and organizing the structure of the knowledge graph. This stage is characterized by its division into two main steps: creating nodes and creating relationships.

- **Creating Nodes:** At this step, the focus was on creating the nodes, which represent the previously extracted entities, categorized into four main types: diseases, symptoms, risk factors, and patients. The method for creating the nodes can be summarized through the following pseudo-code, which organizes the building process in a systematic and structured manner:

Pseudo-algorithm 4: Node Creation

Input: Dataset
Output: Nodes of Knowledge graph
START

Create a node of type **Disease** with the name *'Lung Cancer'*

Define a ***list_of_symptoms***:

['Yellow Fingers', 'Anxiety', 'Chronic Disease', 'Fatigue', 'Allergy',
'Wheezing', 'Coughing', 'Shortness of Breath',
'Swallowing Difficulty', 'Chest Pain']

For each symptom in the ***list_of_symptoms***:

Create a node of type **Symptom** with the corresponding name

Define a ***list_of_risk_factors***:

['Smoking', 'Peer Pressure', 'Alcohol Consuming']

For each risk factor in the ***list_of_risk_factors***:

Create a node of type RiskFactor with the corresponding name

END

- **Creating Relationships:** In this stage, the relationships were established between the patients, and their associated symptoms, risk factors, and diagnosis results on the other side. These relationships were built following precise rules to ensure the consistency and correctness of the connections within the knowledge graph. The process of creating these relationships can be summarized using the following pseudo-code, which clearly illustrates the workflow in a simple and coherent manner:

Pseudo-algorithm 5: Relationships Creation

Input: Dataset
Output: Knowledge Graph

START

For each row (record) in the dataset:

- **Create** a node of type **Person** with `id_patient = row.Id_patient`
- **SET** the **age** attribute of the **Person** node to the numeric value of `row.AGE`
- **For each** `risk_factor` in **list_of_risk_factors**:
  - IF `row[risk_factor]` = 1 THEN
    - **MERGE** a **RiskFactor** node named **risk_factor**
    - **CREATE** a **relationship** `(:Person)-[:hasFactor]->(:RiskFactor)`
  - END IF
- For each **symptom** in **list_of_symptoms**:
  - IF `row[symptom]` = 1 THEN
    - MERGE a **Symptom** node named `symptom`
    - **CREATE** a **relationship** `(:Person)-[:hasSymptom]->(:Symptom)`
  - END IF
- IF `row.LUNG_CANCER` = 1 THEN
  - MERGE a **Disease** node named Lung Cancer
  - **CREATE** a **relationship** `(:Person)-[:DIAGNOSED_WITH]->(:Disease)`
- END IF

END

This algorithm represents the general method for building relationships in the knowledge graph, where each patient is associated with a set of symptoms (Symptom), risk factors

(RiskFactor), and a disease diagnosis (Disease), so as shown in the algorithm the patient must have that symptom to establish a relationship between the patient and that symptom, and the same thing applies to risk factor.

Figure 2.5 shows the Knowledge graph structure obtained for only three patients:

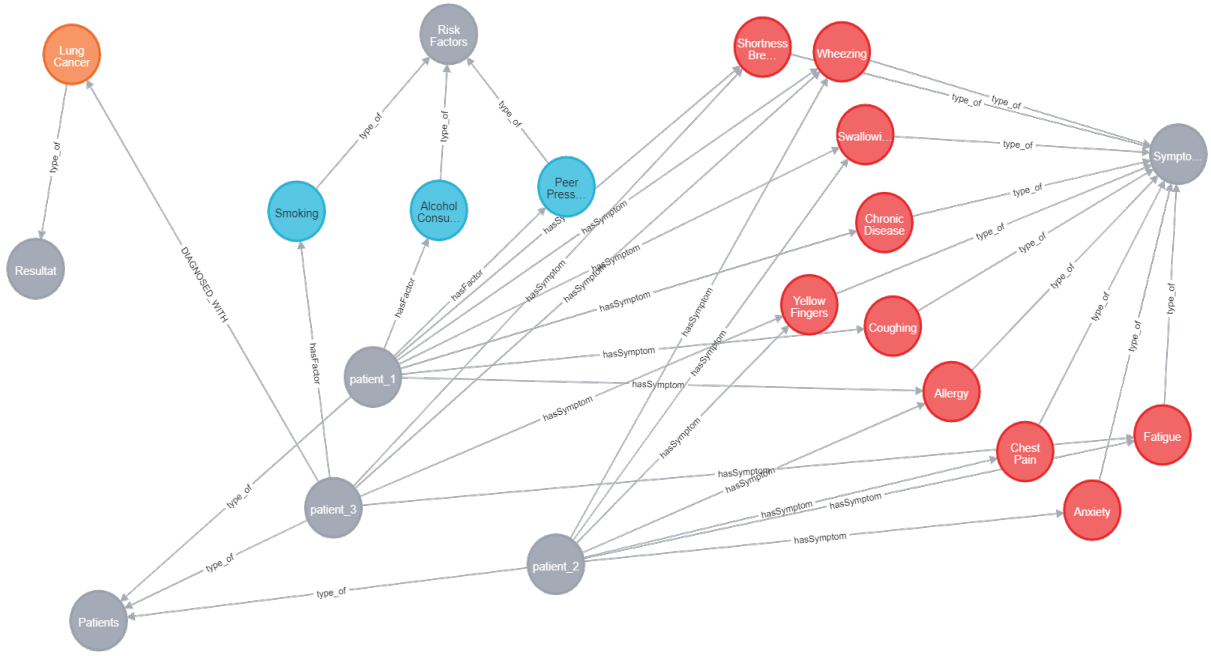

Figure 2.5: Knowledge graph structure

2.4.2 Phase 02: Knowledge Graph Embedding

Knowledge Graph Embedding is the process of transforming the components of a knowledge graph, such as nodes and edges, into numerical representations (vectors) in a low-dimensional continuous space.
The goal of this process is to preserve the original structural and semantic information of the knowledge graph within these numerical representations, making them suitable for processing by deep learning techniques.

2.4.2.1 Using Knowledge Graph Embedding

In this project, we use Knowledge Graph Embedding for several key reasons:

- **Transforming knowledge data into a modeling-compatible format:** After constructing the knowledge graph, which contains information about patients, symptoms, risk factors, and diagnoses, it becomes necessary to convert this information into a numerical form that can be processed using deep learning techniques.

- **Preparation for building the prediction model:** The numerical representations resulting from the embedding process are used as inputs for Graph Neural Networks models, specifically the GAT model adopted in this work.

There are many algorithms used to extract embeddings from knowledge graphs, but we used Node2Vec because it is particularly suitable for directed graphs like ours, and it effectively captures the relationships between nodes when learning embeddings. Node2Vec has been proven to be one of the most efficient and flexible algorithms for network representation learning (Grover Leskovec, 2016).

2.4.2.2 The Principle Followed by Node2Vec for Embedding Extraction

Node2Vec relies on smart sampling of paths (random walks) in the graph, following the principle below:

- **Creation of Directed Random Walks:**
  Node2Vec generates a number of random walks starting from each node.
- **Context Simulation:**
  Each random walk is treated similarly to a text sentence (a sequence of words). Therefore, nodes that appear within the same walk are considered "neighboring" or "contextually related" nodes.
- **Application of Skip-Gram:**
  Node2Vec uses the Skip-Gram model to learn the embedding of each node.
  The goal is for nodes appearing in similar contexts (random walks) to have close numerical representations.
- **Controlling Between BFS (Breadth-First Search) and DFS (Depth-First Search):**
  By adjusting the parameters p and q, Node2Vec can control the nature of the search:
  - A small p value encourages backward moves (local exploration, similar to BFS, It explores the neighboring nodes first (nodes close to the starting point), Think of it as staying shallow and covering all direct connections before moving deeper).
  - A small q value encourages forward moves (deeper exploration, similar to DFS, It explores as deep as possible along one branch before backtracking, Think of it as going deep into one path before trying others).

In other words, Node2Vec does not ignore the relationships between nodes but instead uses them intelligently to generate walks that reflect the connection patterns among entities, thereby building embeddings that capture both the structure of the knowledge graph and the context of its relationships.

2.4.2.3 Node Extraction:

In this step, the different entities of the knowledge graph are identified and extracted. These entities correspond to the main concepts such as patients, diseases, symptoms, and risk factors. Each extracted entity represented numerically through the embedding process.

2.4.2.4 Relationship Extraction:

The relationships linking these entities are also identified. These relationships describe how the nodes are connected (e.g., "patient has symptom"). Preserving the relationships is essential because they capture the structural and semantic meaning of the knowledge graph, which will influence the embeddings.

Pseudo-algorithm: Knowledge Graph Embedding

Input: Knowledge graph
Output: Embedding nodes, Relationship nodes

START

1. Graph projection:
   - Create a graph named **"myGraph"** with nodes of types Person, Symptom, RiskFactor, Disease
2. Embedding computation (vector representations):
   - Apply the Node2Vec method on myGraph
   - Add the embeddings as the `embedding` property to the nodes of the graph
3. Export relationship data:
   - Execute a query to extract all relationships between nodes
   - Export relationships in CSV format with columns:
     - `source` (ID of the source node)
     - `target` (ID of the target node)
     - `relation` (type of the relation)
4. Export node and diagnosis data:
   - Execute a query to retrieve node information including:
     - `node_id` (ID of the node)
     - `node_type` (Person, Symptom, RiskFactor, Disease)
     - `embedding` (vector representation)
     - `diagnosis` (1 if diagnosed with Lung Cancer, 0 otherwise)
   - Export collected data to a **CSV** file

RETURN Embedding nodes, Relationship nodes
END

The figure 2.6 represents an excerpt from the relationship file generated from the constructed knowledge graph. Each row describes a connection between two nodes, where "source" corresponds to the ID of the starting node (for example, a patient) and "target" corresponds to the ID of the destination node (such as a risk factor, symptom, or

disease). The "relation" column specifies the type of link, such as hasFactor, hasSymptom, or DIAGNOSED_WITH. For instance, node 16 is connected to nodes 11 through the hasFactor relation, indicating the presence of risk factors.

source	target	relation
16	11	hasFactor
16	1	hasSymptom
16	4	hasSymptom
16	6	hasSymptom
16	8	hasSymptom
16	0	DIAGNOSED_WITH

Figure 2.6: Example of the extracted Relationships of one Patient Node in the Knowledge Graph

Here is an example of one of the extracted contract inclusion:

node_id	node_type	embedding	diagnosis
14	Person	[-0.5873796,-0.27688187,-0.18715346,0.3211578,-0.19551547,-0.19121869,0.05047115,0.19726549,0.6656192,0.18353641,0.23315679,0.010241227,0.42115894,0.6762436]	1
15	Person	[-0.5975745,-0.20444064,-0.5357867,0.3457073,-0.3901113,-0.032848276,-0.14743537,-0.07139437,0.6161767,0.06720976,0.4109464,0.21988182,0.22198556,0.61256874]	1

Figure 2.7: Example of Person Node Embeddings with Diagnosis Annotations.

The image presents a fragment of the dataset resulting from the construction of the knowledge graph and the application of the Node2Vec algorithm for node embedding. Each entry corresponds to a patient represented by a unique node identifier (node_id), a node type (Person), a list of numerical embeddings, and a diagnosis label. The embeddings are dense vector representations that capture the structural and relational properties of the patient within the graph, reflecting connections to symptoms, risk factors, and diseases. These embeddings are crucial for enabling machine learning models to perform predictive tasks. In particular, the first patient (node_id = 14) has a diagnosis value of 1, indicating a confirmed case of lung cancer. This table illustrates the transition from symbolic medical knowledge into a numerical format suitable for advanced analytical models, thus playing a critical role in the overall pipeline of the proposed solution.

2.4.3 Phase 3: Diagnosis Task

To perform the Diagnosis task in our project, we employed the Graph Attention Network, a type of graph neural network that incorporates attention mechanisms to weigh the importance of neighboring nodes during feature aggregation.

2.4.3.1 Choice of the GAT model for Diagnosis

To justify the choice of the GAT model for diagnosis, Table 2.3 presents a clear comparison between the GAT and GCN:

Factor	**Graph Attention Network**	**Graph Convolutional Network**
Weighting of neighbors	Learns attention weights dynamically; assigns importance to each neighbor	Uniform weight or fixed based on degree
Expressiveness	More flexible, especially in heterogeneous graphs	Simpler but less adaptive
Suitability for XAI	Attention weights can be visualized, aiding interpretability	Less interpretable by design

Table 2.3: Comparison between GAT and GCN

Figure 2.8 illustrates the architecture of the GAT network used, as well as the forward and backward propagation flow for model optimization

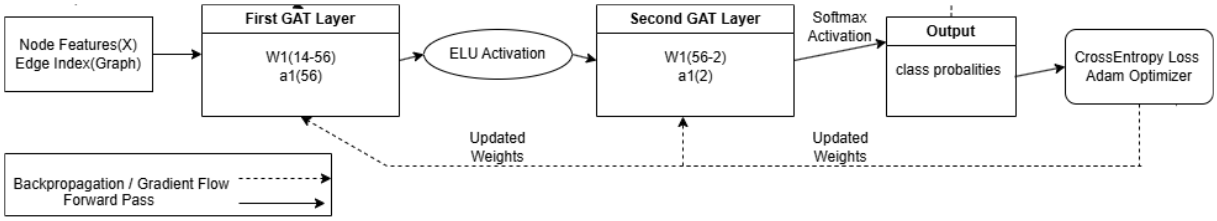

Figure 2.8: Graph Attention Network Architecture with Forward and Backward Propagation Flow

The following algorithm outlines the training process of the Graph Attention Network used in our system:

Pseudo-algorithm: Train a 2-Layer GAT Model

Input: NodeFeatures, Graph (Edge List), Labels
Output: Trained GAT Model

START

Initialize weights `W1`, `W2`
Initialize attention vectors `a1`, `a2`
For each `epoch`:

- **First GAT Layer**:
  - For each `node` in the graph:
    - Look at all its neighbors
    - Compute attention scores between the node and each neighbor
    - Weight neighbor features by attention scores
    - Combine them into a new feature vector

    END FOR
  - Apply ELU activation to all node features → `HiddenFeatures`
- **Second GAT Layer**:
  - For each `node`:
    - Look at its neighbors using `HiddenFeatures`
    - Compute new attention scores
    - Combine neighbor features using new attention

    END FOR
  - Result → `FinalOutputs`
- **Classification**:
  - Apply Softmax to `FinalOutputs` to get class probabilities
- **Loss and Update**:
  - Compute CrossEntropy loss between predictions and `Labels`
  - Backpropagate the loss
  - Update `W1`, `W2`, `a1`, `a2` using the optimizer

END FOR
END

2.4.3.2 GAT Model Process (Forward Pass)

1. **Loading Node Features and Graph Structure**
Our model begins by loading:

- **Node features** (embeddings), which represent each entity in the graph such as patients, symptoms, or disease markers.
- **Graph structure** (edges), which defines how nodes are connected (e.g., "patient has symptom", "symptom indicates disease").

2. **First GAT Layer – Attention-Based Aggregation**
Each node aggregates information from its neighboring nodes using a self-attention mechanism.
For each pair of connected nodes i and j, attention scores α_{ij} are computed to determine how much influence node j should have on node i.

3: Multi-Head Attention and Concatenation

In our model, 4 attention heads are used. Each head computes its own set of attention coefficients and node embeddings. Their outputs are concatenated to capture different types of relationships. For example, if each head outputs a 16-dimensional vector, the result is a 64-dimensional vector per node.

4: Applying ELU Activation

The ELU (Exponential Linear Unit) activation function is applied to the concatenated outputs to introduce non-linearity:

$$\text{ELU}(x) = \begin{cases} x, & \text{if } x \geq 0 \\ \alpha(\exp(x) - 1), & \text{if } x < 0 \end{cases}$$

Here, α is a positive parameter that controls the slope of the function for negative values of x. This helps the model capture complex patterns in the data.

5: Second GAT Layer – Final Aggregation

The second GAT layer takes the 64-dimensional embeddings and applies attention-based aggregation again, but with only one head. Unlike the first layer, this one outputs a 2-dimensional vector per node (the logits), without concatenation.

6: Output – Raw Logits

The output of the second layer consists of raw logits for two classes: disease or no disease. These logits are passed into the loss function for classification.

2.4.3.3 Training Process (Backward Pass and Optimization)

1: Set Model To Train Mode

This enables gradient computation and, if present, activates dropout layers.

2: Forward Pass

The model computes predictions using the forward method described above.

3: Compute Loss

Cross-entropy loss is calculated using only the nodes in the training mask. The function internally applies softmax to the logits and compares them with true labels.

4: Backward Pass

The `.backward()` function computes gradients of the loss with respect to model parameters.

5: Update Parameters

The optimizer (Adam) updates model weights using the computed gradients to reduce the loss.

6: Zero Gradients

Gradients are cleared to prepare for the next iteration.

2.4.4 Phase 4: Model Interpretation

This phase represents a critical component of the project, as it involves interpreting the model's predictions. Our focus is on post-hoc interpretability providing explanations after the prediction has been made. We utilize LIME to generate local explanations, which help us understand the model's behavior on individual predictions. In contrast, SHAP is employed for global interpretability, offering a broader view of how each feature contributes to the model's predictions across the entire dataset. The combination of local and global explanations ensures a comprehensive understanding of the model's decision-making process, both at the individual instance level and at the overall model level.

2.4.4.1 Process Of LIME Explanation:

LIME is a technique used to explain individual predictions of any machine learning model by approximating it locally with an interpretable model (usually linear).

It works by generating perturbed samples around a data point and observing how the model's prediction changes. It then learns which features are most responsible for the prediction for that specific instance.

We chose LIME because it explains individual predictions, which is essential in clinical settings. It highlights the most important features in a way that's easy to understand, helping clinicians see if the model's reasoning aligns with known medical knowledge. This builds trust and supports better decision-making.

Pseudo-algorithm: LIME Explanation process

Input:

- A table with medical data for many people (e.g., age, smoking, alcohol use...)
- A trained model that predicts if a person has lung cancer or not
- The data for one specific person you want to explain

Output: A clear list of top features (like "smoking", "age", etc.) that explain why the model said the person has (or doesn't have) lung cancer.

START

1. Create similar people:
    - LIME creates many imaginary versions of this person by changing their data slightly.
2. Send all fake people to the model:
    - LIME sends all the new fake people into the same model you trained.
    - For each fake person, it gets a prediction (e.g., lung cancer or not, with a confidence score)
3. Watch what changes affect predictions:
    - LIME looks at which changes in the data made the model's output change the most:
4. Build a simple explanation:
    - LIME builds a very simple and understandable model just for this person.
    - It copies how the real model behaves around that person but uses only the important features.
5. Identify the top features:
    - From that simple model, LIME finds which medical factors mattered most in the final decision.
6. Identify the top features:
    - LIME gives a list or chart of the most influential factors and how strongly they affected the model's prediction.

END

Explaining Individual Predictions Using LIME

1: Input Preparation

We extracted the original patient features (e.g., age, smoking history, fatigue, etc.) from the dataset while excluding identifiers and labels. This set of interpretable features serves as the basis for explanation.

The label column (e.g., `LUNG_CANCER`) was excluded because it represents the outcome the model is trying to predict. Including it as an input feature would violate the integrity of the explanation process and result in circular reasoning. This ensures that LIME explanations are based solely on interpretable, relevant input features that the model actually used for its decision-making.

2: Black-box Model Definition

To allow LIME to interpret predictions made by our GAT model, we defined a custom prediction function. This function accepts the tabular clinical features as input and internally integrates them into the graph structure before forwarding them through the GAT model. This setup treats the GAT model as a black box, enabling compatibility with LIME.

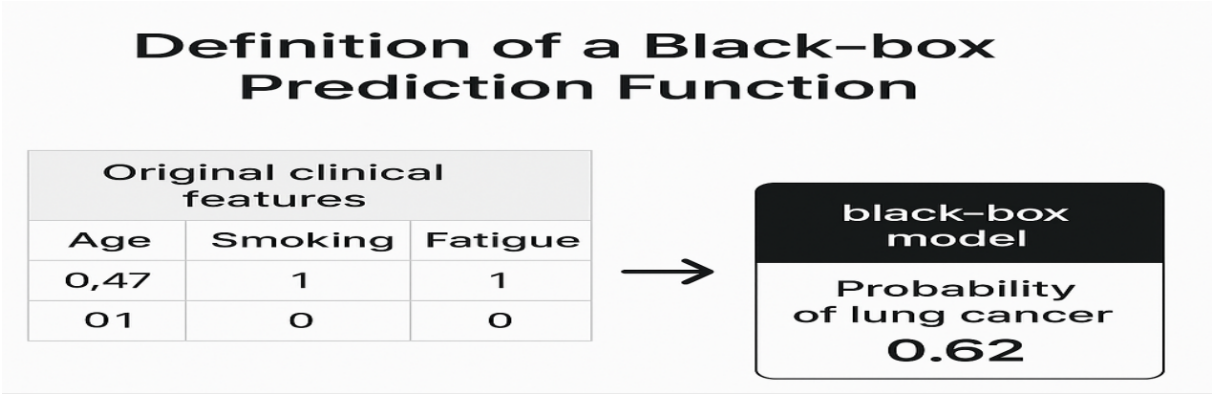

Figure 2.9: Definition of a Black-box Prediction Function

3: Explainer Initialization

We initialized the LIME explainer using the training feature matrix. Key parameters defined here include:

Hyperparameter	Value	Description
training_data	Original tabular features	Reference data used to generate similar samples.
feature_names	Clinical feature names	Used to display interpretable outputs.
class_names	["No Lung Cancer", "Lung Cancer"]	Human-readable output classes.
discretize_continuous	True	Converts continuous values into intervals to simplify interpretation.
mode	Classification	Indicates that the task is a classification problem.

Table 2.4: LIME Explainer Initialization Parameters

4: Instance-Level Explanation

We selected a specific patient instance and asked LIME to explain the prediction for that individual. LIME perturbed the input values slightly to create similar synthetic samples, passed each one through the GAT model, and trained a simple surrogate model (typically linear) to approximate the local decision boundary. Key parameters in this step:

Parameter	Value	Description
num_features	14	Number of top influential features to include in the explanation.
num_samples	10 (or more)	Number of synthetic samples generated around the instance.

Table 2.5: Key Parameters for LIME Explanation

5: Output Interpretation

LIME returned a ranked list of features, indicating whether each one increased or decreased the likelihood of predicting lung cancer for that patient. These results were visualized in an interpretable format for end-user understanding (e.g., clinicians).

In general, the LIME explanation shows how the model made its prediction for one specific patient by highlighting which features had the biggest influence. It splits the factors into two groups:

– On one side, the features that lower the risk of lung cancer.

– On the other side, the features that raise the risk.

The size of each bar shows how important that feature was in the decision. So, we can quickly see which health indicators pushed the prediction toward “Lung Cancer” and which ones pushed it toward “No Lung Cancer.” This helps make the model’s reasoning more understandable and transparent.

2.4.4.2 Process Of SHAP Explanation:

SHAP is chosen for global interpretability due to its ability to rank features by their overall influence on model predictions across all patients. Grounded in cooperative game theory through the use of Shapley values, SHAP provides a theoretically sound and fair approach to feature attribution. It complements LIME by going beyond local explanations to offer insights into which features consistently impact predictions throughout the dataset. This makes SHAP particularly valuable for identifying key risk factors—such as smoking and fatigue—that are influential across the entire patient population, thereby enhancing trust and understanding in the model's behavior.

Pseudo-algorithm: SHAP Explanation process

Input:

- A trained GAT model for lung cancer prediction
- Patient feature data X (without labels or IDs)
- A background sample (e.g., 100 patients)
- A sample to explain (e.g., 2998 patients)

Output: A ranked list of medical features showing:

- Their average importance in predicting lung cancer
- Their direction (whether they increase or reduce risk)

START

1. Choose a reference (background):
    - Select a small random group of patients (e.g., 100 people This is the baseline that SHAP uses to compare all other patients).
2. Select the people to explain:
    - Pick many people from the dataset (e.g., 2998 patients. These are the ones whose predictions you want to interpret).

   For each person, SHAP does the following:

3. Simulate feature removals (perturbations): For each person, SHAP creates many versions of them by:
    - Hiding or turning off different features like "smoking", "age", etc. (These are not fake people but feature combinations used to understand impact).
4. Run these versions through the model: Each modified version is:
    - Scaled (normalized)
    - Inserted into the graph as updated patient nodes
    - Passed into the GAT to get new predictions

   SHAP observes how the prediction changes each time

Pseudo-algorithm: SHAP Explanation process

5.Measure the effect of each feature:

- SHAP compares predictions with and without each feature. (If removing a feature changes the output a lot → it's important If the output doesn't change → it's not important).

END FOR

6.Display the results:

- SHAP creates a summary plot that shows: (Which features matter most across all people Whether each feature increases or decreases cancer risk).

END

Component	**Description**
Input features	Original clinical variables (e.g., age, smoking, fatigue)
Prediction output	Lung cancer probability (continuous score)
Background dataset	A subset of typical patient records (used as a reference baseline)
Explained instances	A large sample of patients (used to generalize feature importance globally)
Samples per instance	Number of perturbed samples used to estimate feature contributions

Table 2.6: Inputs and Hyperparameters for SHAP Analysis

2.5 Conclusion

Throughout this chapter, we presented the detailed steps of our proposed solution to interpret the decisions of deep learning models based on knowledge graphs. We discussed four main stages: constructing the knowledge graph, generating graph embeddings, building a deep learning model, and applying explainable AI techniques to interpret the model's outputs. In the following chapter, we will implement the proposed design in practice, conduct experiments, and evaluate the obtained results.

Chapter 3

System Implementation and Testing

3.1 Introduction

After completing the design phase of our system, this chapter focuses on the practical aspect of its implementation. We begin with a presentation of the development environments and tools used, followed by the system's validation and testing phases. Finally, a visual illustration of our application will be provided through screenshots showcasing the different system interfaces.

3.2 Programming Language

Python: Python is one of the most popular modern programming languages, known for its simplicity and readability. It is widely used across various fields, such as web and desktop application development, data science, and artificial intelligence. Python offers a wide range of ready-to-use libraries, such as **NumPy** and **Pandas** for data processing, and **Flask** for web application development, which makes it easier for developers to implement complex projects efficiently. Additionally, Python is supported by a large and active developer community, contributing to the availability of abundant learning resources and continuous technical support. In my project, I chose Python for its rapid development capabilities, flexibility in modification, and scalability, making it a powerful and efficient tool for building complete software applications in a short time.

3.3 Material Used

Work and tests were carried out on:

- Processor: Intel Core i5-1145G7 @ 2.60 GHz, 8 CPUs
- RAM: 16 GB

- OS: Windows 11 Professional 64-bit

3.4 Software Used

- **VS Code:** Visual Studio Code is one of the most popular open-source integrated development environments (IDEs), widely used for developing various types of applications. It is known for its lightweight design, speed, and extensive support for programming languages through its powerful extension system. In this project, VS Code was used as the primary environment for writing and editing source code in Python, providing a flexible, comfortable, and well-organized workspace.

- **Draw.io:** Draw.io is a free and user-friendly tool widely used for creating diagrams and visualizations. It offers an intuitive interface and a vast library of graphic elements that can be dragged and dropped to build flowcharts, diagrams, mockups, and other visual representations.

- **Neo4j:** Neo4j is a graph-oriented database management system designed to store data as nodes and relationships rather than traditional tables. It is particularly well-suited for applications that require modeling complex relationships between entities, such as social networks or recommendation systems. In our project, Neo4j was used to build the knowledge graph, which allowed us to represent and query interconnected data more effectively. Thanks to its use of the Cypher query language, Neo4j enabled high-performance data operations and intuitive handling of complex graph-based structures. .

3.5 Test Procedure

To effectively evaluate the efficiency of our system, we performed a validation of our knowledge graph. Then, we compared the performance of two different models using two algorithms for knowledge graph embedding extraction according to the following procedure:

3.5.1 Step 1: Knowledge Graph Validation

Firstly, it is important to note that the knowledge graph was manually constructed with a high level of precision, in order to ensure a comprehensive and well-structured representation that meets all domain-specific requirements. To ensure the reliability and completeness of the constructed knowledge graph, we evaluated its quality based on four key semantic and structural metrics: Logical Relations, Consistency, ICR (Instantiated Class Ratio) and IPR (Instantiated Property Ratio) metrics.

Metric	Definition
1. Logical Relations Metric	This metric evaluates the logical coherence and correctness of relationships between entities in the knowledge graph(17).
2. Consistency Metric	This metric evaluates the degree to which the knowledge graph is free from logical contradictions or inconsistencies between entities and relationships(18).

Table 3.1: knowledge graph quality evaluation metrics

A. Logical Relations Metric

The objective of this metric is to verify that the relationships between entities are logically coherent and semantically meaningful.

- **Relation Analysis in the Graph:**

Relationship	**Source → Target**	**Is it logical?**
hasSymptom	Person → Symptom	Yes
hasFactor	Person → RiskFactor	Yes
DIAGNOSED_WITH	Person → Disease	Yes

➤ After analyzing the knowledge graph relationship, we recorded the following observations:

➢ No illogical inverse relationships were found (e.g., Symptom → Person).
The logical relations are 100% correct.

➢ No ambiguous or generic relationships were used .

➤ We conclude that logical structure of the graph is sound, with 100% valid relationships.

B. Consistency Metric

The objective of this metric is to determine whether the graph contains internal contradictions or misuses of relationships.

➤ **Consistency Analysis:**

➢ **The use of relationships is coherent:**
hasSymptom and hasFactor correctly connect to Person nodes.
DIAGNOSED_WITH correctly links the patient to the disease.

➢ **No contradictions such as:**
A person being associated with two contradictory properties (e.g., both male and female).
A relationship in the wrong direction (e.g., Disease → hasSymptom → Person).

➤ We conclude that graph is constructed with clear logic and free from contradictions with **High Consistency**.

C. Structural Quality Metrics

-Instantiated Class Ratio (ICR)

Instantiated Class Ratio is a metric that measures the extent to which classes defined in the ontology are actually used in the knowledge graph. It is calculated as the ratio of the number of classes that have actual instances to the total number of defined classes. The minimum acceptable value for ICR is 0.5 (16).

$$\text{ICR} = \frac{N(IC)}{N(C)}$$

where:

- $N(IC)$: Total number of defined classes in the ontology
- $N(C)$: Number of classes that have actual instances in the graph

The ontology implemented in the knowledge graph defines four main classes:

  - Person
  - Symptom
  - RiskFactor
  - Disease

All of these classes are actively instantiated within the knowledge graph through the use of MERGE commands in the Cypher script, where actual instances are created.
Accordingly, the ICR metric is calculated as follows:

$$\textbf{ICR} = \frac{4}{4} = 1 \cdot$$

- A perfect ICR score of 1.0 indicates that:
  ➢All classes defined in the ontology are actually utilized within the knowledge graph, and none remain unused.
  ➢This reflects a strong alignment between the conceptual model (ontology) and the actual data, which is a key factor in enhancing the effectiveness of the knowledge graph in reasoning tasks.

-Instantiated Property Ratio (IPR)

Instantiated Property Ratio measures how well the defined properties (either relationships or attributes) in the ontology are utilized in the actual data of the knowledge graph. It

is computed as the ratio of used properties to total defined properties. The minimum acceptable value for IPR is 0.6 (16).

$$\text{IPR} = \frac{N(IP)}{N(P)}$$

where:

- $N(IP)$: Number of properties actually used in triples or node attributes.
- $N(P)$: Number of defined properties.

The ontology implemented in our knowledge graph includes the follows properties:

- Object Properties (Relationships):
  - hasSymptom
  - hasFactor
  - DIAGNOSED_WITH
- Data Property (Attribute):
  - age: a literal attribute assigned to each Person instance, representing the patient's age.

The ontology, therefore, defines a total of four properties, all of which are effectively instantiated in the knowledge graph through connections or attributes in the data.

$$\textbf{IPR} = \frac{4}{4} = 1 \cdot$$

- A perfect IPR score of 1.0 indicates that:
  ➢ All defined properties are actually used in representing knowledge.
  ➢ There are no unused or redundant relationships, pointing to a well-designed and purpose-driven ontology.

-The ideal scores of ICR = 1.0 and IPR = 1.0 strongly indicate a high structural quality in the knowledge graph. The ontology is both well-designed and fully utilized, ensuring consistency between the conceptual model and the data it represents.

3.5.2 Step 2: Model Validation Through Performance Measures

We used the confusion matrix to summarize the correct and incorrect predictions made by our models, Table 3.2 presents the general structure of the confusion matrix:

	Predicted: True	**Predicted: False**
Actual: True	TP	FN
Actual: False	FP	TN

Table 3.2: Confusion Matrix Structure

➤TP – True Positive: The number of instances that are actually positive and the model correctly predicted them as positive.
➤TN – True Negative: The number of instances that are actually negative and the model correctly predicted them as negative.
➤FP – False Positive: The number of instances that are actually negative, but the model incorrectly predicted them as positive.
➤FN – False Negative: The number of instances that are actually positive, but the model incorrectly predicted them as negative.

This step involves evaluating two different models (GAT and GCN) that we developed, with selecting the one that offers the best performance based on the following four evaluation metrics:

A. Accuracy: Accuracy is one of the fundamental metrics for evaluating the performance of classification models. It represents the percentage of correct predictions over the total number of instances. It is especially useful when the dataset is balanced.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FN + FP}$$

B. Precision: Precision measures the proportion of predicted positive instances that are truly positive. It reflects the accuracy of the positive predictions made by the model.

$$\text{Precision} = \frac{TP}{TP + FP}$$

C. Recall: Recall measures the proportion of actual positive instances that were correctly identified by the model. In other words, it evaluates the model's ability to retrieve all relevant positive cases.

$$\text{Recall} = \frac{TP}{TP + FN}$$

D. F1-Score: The F1-score is a harmonic mean of Precision and Recall, providing a single metric that balances both. It is particularly effective in cases of imbalanced datasets where accuracy alone may be misleading.

$$F\text{-score} = \frac{2 \cdot (\text{Précision} \cdot \text{Rappel})}{\text{Précision} + \text{Rappel}}$$

Regarding model selection, we used two knowledge graph embedding algorithms, which serve as input for the developed models: Node2Vec and Trans E (Translating Embeddings

for Modeling Multi-relational Data).

Trans E is an algorithm proposed by Antoine Bordes et al. in 2013, in a paper published under the same title. TransE is one of the first and most widely used embedding models for knowledge graphs, based on the idea of translating relationships between entities as vectors in a latent space.
This table 3.2 presents the results of the Accuracy, F1-Score, Precision and Recall metrics for the two models, based on the knowledge graph embedding algorithms used.

Model	**Algorithm**	**Accuracy**	**F1-Score**	**Precision**	**Recall**
GAT	Node2Vec	0.8027	0.7878	0.8056	0.7708
	Trans_E	0.6401	0.6226	0.6581	0.5908
GCN	Node2Vec	0.7529	0.7496	0.7360	0.7637
	Trans_E	0.6036	0.6088	0.6019	0.6159

Table 3.3: Summary of Evaluation Metrics for GAT and GCN

To better illustrate and compare the results presented in the previous table, we have graphically represented the performance of the GAT and GCN models using the two embedding extraction algorithms: Node2Vec and TransE. This chart highlights the performance differences between the tested combinations in terms of Accuracy, F1-Score, Precision, and Recall, facilitating visual interpretation of the results and reinforcing the quantitative analysis.

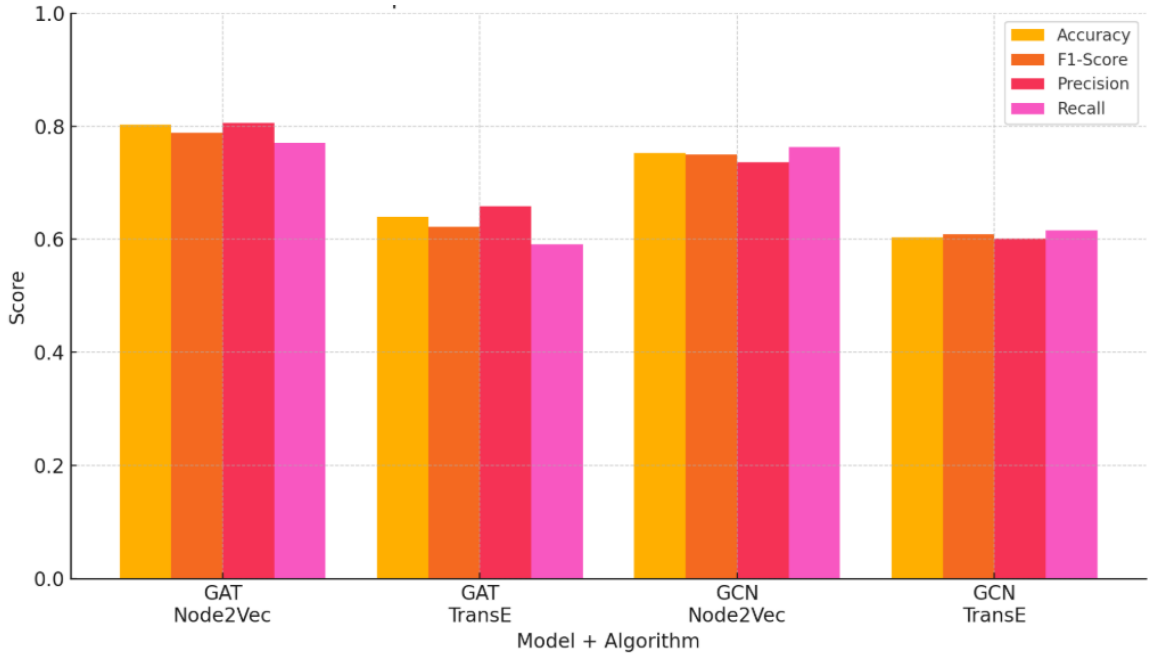

Figure 3.1: Comparative Performance of GAT and GCN with Node2Vec and

➤ **Comparative Analysis of Results**

The table and chart above present the performance of the GAT and GCN models using two knowledge graph embedding algorithms: **Node2Vec** and **TransE**. The evaluation is based on four metrics: *Accuracy*, *F1-Score*, *Precision*, and *Recall*.

Overall, the results show that **Node2Vec** outperforms **TransE** regardless of the model used. For instance, with the GAT model, Node2Vec achieves an **accuracy** of 0.8027, an **F1-Score** of 0.7878, a **precision** of 0.8056, and a **recall** of 0.7708. In contrast, these values drop significantly when using TransE (accuracy = 0.6401, F1 = 0.6226, precision = 0.6581, recall = 0.5908).

A similar trend is observed with the GCN model, where Node2Vec again yields better results across all metrics (accuracy = 0.7529, F1 = 0.7496, precision = 0.7360, recall = 0.7637) compared to TransE (accuracy = 0.6036, F1 = 0.6088, precision = 0.6019, recall = 0.6159).

When comparing **GAT** and **GCN**, it is clear that GAT combined with Node2Vec provides the best overall performance. This can be attributed to GAT's ability to incorporate *attention mechanisms* in the aggregation of information from neighboring nodes, thereby enhancing prediction accuracy.

➢ In our case, we obtained Table 3.4, which presents the confusion matrix of our model (GAT + Node2Vec).

	Predicted: True	**Predicted: False**
Actual: True	246	56
Actual: False	64	237

Table 3.4: Confusion Matrix

The confusion matrix presented in Table 3.4 provides insight into the performance of the classification model. Out of the total samples, 246 instances were correctly classified as True (true positives), while 237 were correctly classified as False (true negatives). However, the model misclassified 56 actual True instances as False (false negatives) and 64 actual False instances as True (false positives). These results indicate that the model demonstrates a strong ability to distinguish between the two classes, achieving high accuracy. Nevertheless, the presence of both false positives and false negatives suggests there is still room for improvement, particularly in enhancing sensitivity and specificity, depending on the criticality of each type of error in the medical context.

3.6 Interpretation of GAT Model Prediction Results

After obtaining the predictions from the GAT model, and applied two model explanation techniques to better understand its decision-making process. We used LIME to generate local explanations, providing insight into how individual features influenced specific predictions. On the other hand, SHAP was applied to analyze the model's behavior at a global level, identifying the most influential features across the entire dataset. In the following section, we analyze and interpret the results of these two methods.

3.6.1 Interpretation with SHAP (global-explanation)

3.6.1.1 Bar Chart of SHAP Values of model Prediction

The Figure 3.2 shows how features impacted the GAT model prediction (*global explanation*). Features in red increase the predicted probability (**positive impact**), and features in blue decrease the predicted probability (**negative impact**).

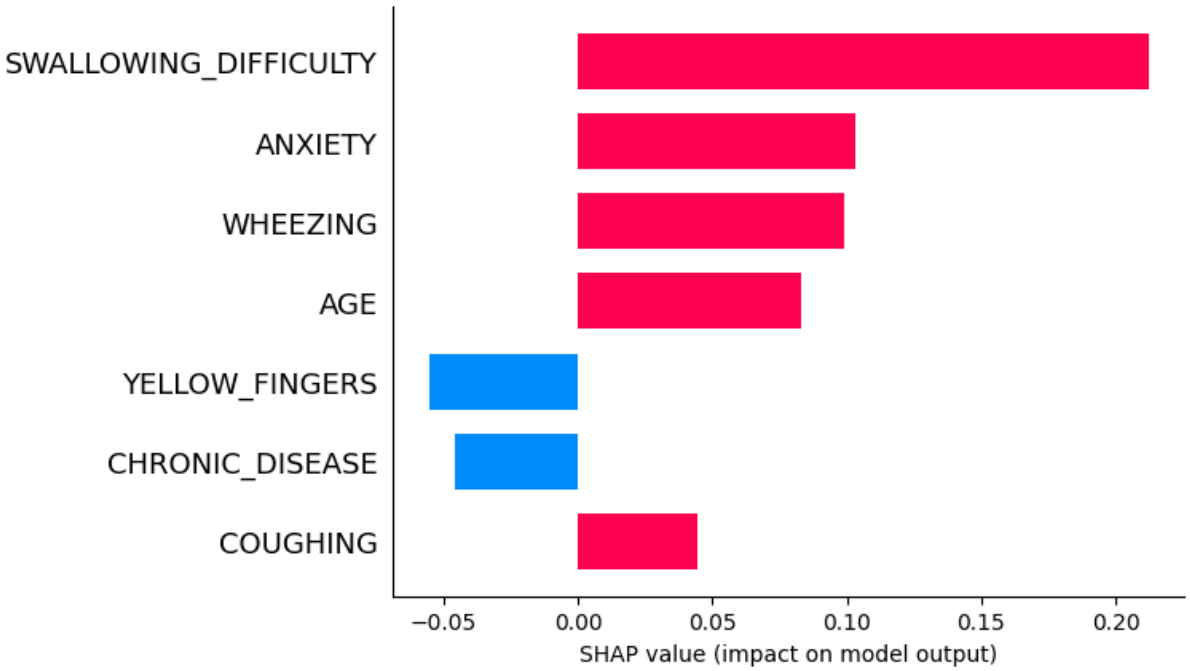

Figure 3.2: Contributing Features of the GAT model prediction (global SHAP Explanation)

We observe that: `SWALLOWING_DIFFICULTY`, `ANXIETY`, `WHEEZING`, and `AGE` had the strongest **positive** effect on this prediction, while `ALLERGY` and `CHRONIC_DISEASE` had the highest **negative** influence.

3.6.1.2 Raw feature Importance Values

The figure **3.3** shows a list of the mean absolute SHAP values for each feature sorted by importance.

```
SMOKING                  : 0.2179
ALLERGY                  : 0.1852
SHORTNESS_OF_BREATH      : 0.1260
SWALLOWING_DIFFICULTY    : 0.1253
WHEEZING                 : 0.0896
AGE                      : 0.0787
CHRONIC_DISEASE          : 0.0658
ANXIETY                  : 0.0537
COUGHING                 : 0.0418
FATIGUE                  : 0.0390
YELLOW_FINGERS           : 0.0259
ALCOHOL_CONSUMING        : 0.0107
CHEST_PAIN               : 0.0094
PEER_PRESSURE            : 0.0036
```

Figure 3.3: Feature Importance from Aggregated SHAP Values (Numerical Summary)

We observe that the top features are:
SMOKING (0.2179), ALLERGY (0.1852), SHORTNESS_OF_BREATH (0.1260), and SWALLOWING_DIFFIC (0.1253). This provides a *global perspective*: higher values indicate that a feature has a greater average impact on the model's output across all predictions. This summary is global and helps determine which features the model relies on most.

3.6.1.3 Raw feature Importance Values

The Figure 3.4 shows how each individual feature influenced the prediction outcome for each sample in the dataset. Rows represent individual samples (e.g., patients or observations).

The color of each cell reflects the SHAP value of that feature for that sample:

- **Red (Positive SHAP value):** The feature increased the model's predicted probability for the positive class.
  → It had a **positive impact** on the prediction.
- **Blue (Negative SHAP value):** The feature decreased the predicted probability.
  → It had a **negative impact** on the model's output.

- **White or light colors (SHAP ≈ 0):** The feature had little to no influence on the prediction for that sample.

Figure 3.4: Per-Sample SHAP Impact Heatmap Across All Features

We observe that:
SMOKING, ALLERGY, SHORTNESS_OF_BREATH, and SWALLOWING_DIFFICULTY show strong consistent patterns of influence, while features like PEER_PRESSURE or ALCOHOL_CONSUMING are mostly light or neutral, confirming their low influence on the prediction.

3.6.1.4 SHAP Summary Plot

The Figure 3.5 ranks features based on their overall impact on the model output.

- The **x-axis** shows SHAP values (impact on prediction).
- Each dot represents a SHAP value for a feature in one sample.
- Color represents the feature value: red = high, blue = low.

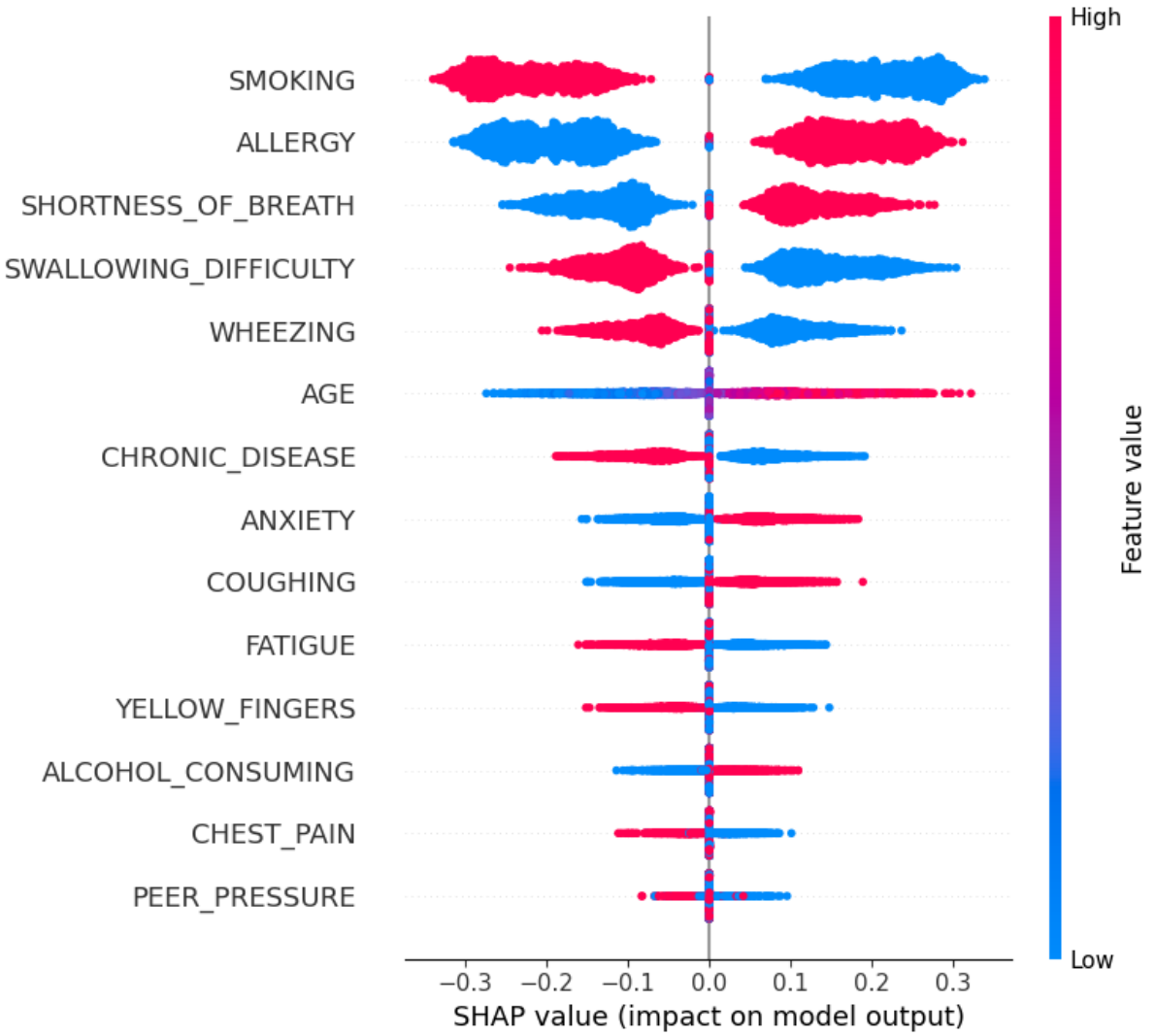

Figure 3.5: Global Feature Impact Summary Using SHAP

We observe that:

- SMOKING, ALLERGY, and SHORTNESS_OF_BREATH have the highest average influence. For example, higher SMOKING values (in red) mostly push predictions positively (to the right).
- Features like ALCOHOL_CONSUMING, peer pressure, and chest pain have little impact across all samples.

3.6.2 Interpretation with LIME(local-explanation)

3.6.2.1 LIME Feature Contribution Visualization (Color- Bars)

This horizontal bar chart in Figure 3.6 shows the individual contribution of each feature toward the model's prediction. The **blue bars** represent features that **negatively influenced** the prediction (i.e., pushed the model towards the negative class or reduced the likelihood of the predicted class). The **orange bars** indicate features that **positively influenced** the prediction (i.e., increased the probability of the predicted class). This helps us identify which features had the most significant influence on the model's decision locally (for a single prediction).

Figure 3.6: Graphical Overview of Local Feature Contributions (LIME)

We observe that the **Top Contributing Features** are:

- COUGHING (strongest negative influence)
- WHEEZING (positive)
- ALCOHOL_CONSUMING and FATIGUE (negative)
- SWALLOWING_DIFFICULTY and SMOKING (positive)

3.6.2.2 LIME Feature Importance with Numerical Weights

This view lists the top features numerically with their respective LIME contribution scores. Positive values mean the feature increased the prediction score, while negative values decreased it. This provides a precise, ranked list of feature contributions, which is useful for quantitative interpretation and debugging. The sign and magnitude of each value directly indicate how much and in which direction the feature influenced the model's decision.

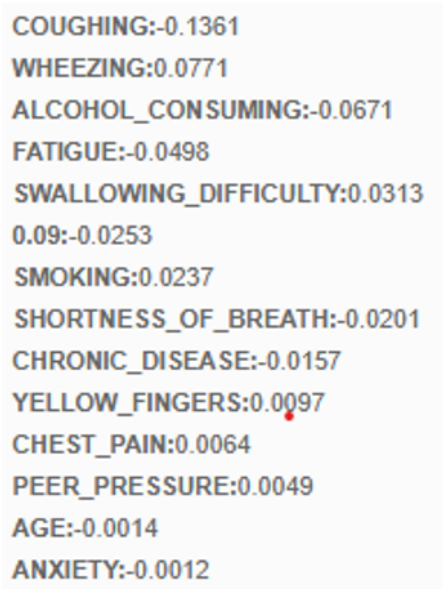

Figure 3.7: Ranked Feature Importance Scores for Local Prediction (LIME Output)

We observe that the **Top Positive Influences** are:

- WHEEZING: +0.0771
- SWALLOWING_DIFFICULTY: +0.0313
- SMOKING: +0.0237

And the **Top Negative Influences** are:

- COUGHING: -0.1361
- ALCOHOL_CONSUMING: -0.0671
- FATIGUE: -0.0498

3.7 System Testing and Exploitation

After completing the development of the intelligent medical prediction and decision explanation system, we tested it in collaboration with a medical expert. The goal was to

evaluate its effectiveness, and verify their consistency with real medical expertise. The system's performance was analyzed through:

Prediction Test on a Real Medical Case

We input an anonymized medical case file into the system to generate a prediction and its explanation using **LIME**. The result presented to the doctor included:

- **The prediction result**.
- **The LIME explanation**, showing each feature's contribution to the decision and how it influenced the specific prediction for this case.

A. Properties:

- The patient is 50 years old.

B. Risk Factors:

- The patient **smokes**.
- The patient **does not drink alcohol**.
- The patient has **peer pressure**.

C. Symptoms:

- The patient has **no yellow fingers**.
- The patient has **fatigue**.
- The patient has **allergy**.
- The patient has **wheezing**.
- The patient has **no chronic diseases**.
- The patient has **anxiety**.
- The patient has **coughing**.
- The patient has **shortness of breath**.
- The patient has **swallowing difficulty**.
- The patient has **no chest pain**.

After entering the patient's data into the system, we obtained the following result:

- **Predicted probability of lung cancer: 94.5%**

- **Predicted probability of no lung cancer: 5.5%**

Based on these probabilities, we conclude that the model predicts the patient has lung cancer with a high degree of confidence. Additionally, the system provided a detailed explanation of how each feature contributed to the prediction, offering insight into the rationale behind the decision.

The influential features were categorized as follows:

- Features with a positive contribution(increased the probability of cancer): Smoking, Allergy, Swallowing difficulty, Shortness of breath, Wheezing, Anxiety, Fatigue, Coughing, Chest pain.
- Features with a negative contribution (decreased the probability of cancer): Chronic disease, Yellow fingers, Alcohol consuming, Age, Peer pressure.

The quantitative contribution of each feature to the prediction was as follows: This detailed

Feature	**Contribution**
Smoking	+0.1241
Allergy	+0.1159
Swallowing difficulty	+0.0818
Shortness of breath	+0.0749
Wheezing	+0.0613
Chronic disease	-0.0462
Anxiety	+0.0413
Fatigue	+0.0405
Coughing	+0.0306
Yellow fingers	-0.0272
Chest pain	+0.0236
Alcohol consuming	-0.0193
Age	-0.0053
Peer pressure	-0.0041

Table 3.5: Feature contributions to the prediction based on LIME

explanation enhances the system's transparency and allows the medical expert to evaluate the reasoning behind the prediction in light of clinical standards.

3.8 Application Presentation

1. Home page Interface The Figure 3.8 above represents the **Home Page Interface** of our system.

At the top left, the sidebar contains a navigation menu titled **Interpretation**, which includes:

- Accueil: redirects the user to the main dashboard or homepage.
- Prediction (Figure 3.9) and Shap Result (Figure 3.12): Two buttons Navigates to a page where users submitting data, and dedicated page showing SHAP visualizations
- FAQ (Figure 3.13): Frequently Asked Questions section, Provides help.
- FAQ (Figure 3.14): link leads to a page or visualization of the knowledge graph

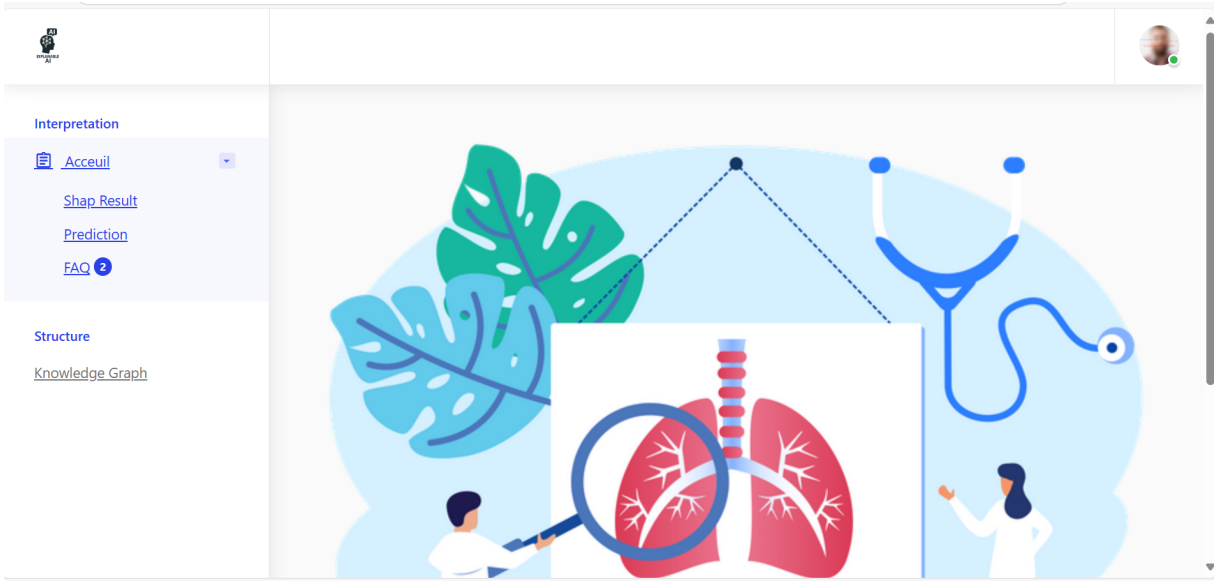

Figure 3.8: home page

2. Interpretation Submission Interface

The Figure 3.9 above represents the **Interpretation Submission Form** of our system. This interface allows the user to input various patient-related data to interpret the model's prediction.

In the center, the main form is titled **Interpretation Submission Form**. It consists of several fields and binary choices (**Yes/No**) for medical attributes:

- AGE: A text input where the user must enter the patient's age.
- SMOKING, YELLOW_FINGERS, PEER_PRESSURE, etc.: Binary choices allowing the user to answer with **Yes** or **No** based on the patient's symptoms or lifestyle factors.

- Tooltips (small icons next to the labels) provide additional explanations for specific attributes such as FATIGUE, WHEEZING, CHRONIC_DISEASE, etc.
- Mandatory fields are indicated by red asterisks.

At the bottom of the form, there is a Submit button that allows the user to send the filled-in data and to see the result of LIME local interpretation.

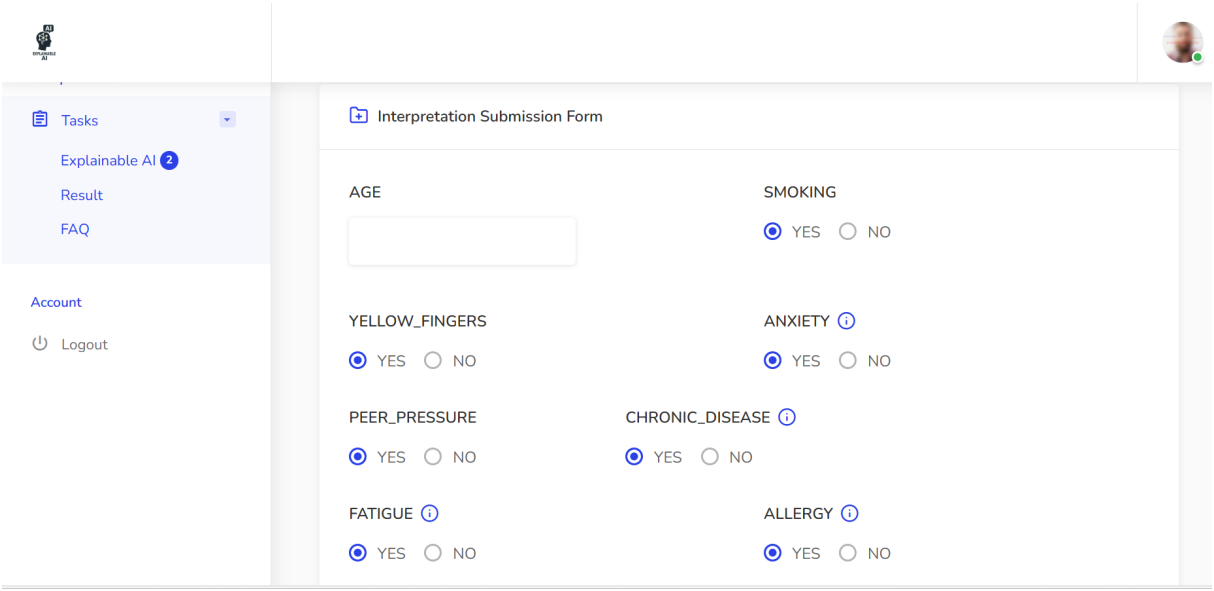

Figure 3.9: Submission Interface

3.LIME Interpretation Result Interface

After submitting the interpretation form in the previous step, the user is redirected to this results interface (Figure 3.10), which displays the prediction outcome along with a visual explanation using LIME.

At the center of the interface:

- A prediction result section is shown at the top, presenting the model's estimated probability:
  - **Cancer:** 93.22%
  - **No Cancer:** 6.78%

  This result is also visualized using a bar chart, where the higher bar highlights the predicted class (Cancer) with its corresponding confidence level.
- Below the chart, the LIME Explanation section breaks down how individual features influenced the prediction:
  - Each row represents a feature (e.g., COUGHING, WHEEZING, ALCOHOL_CONSUMING, etc.).

– Features are color-coded, indicating positive or negative contributions to the cancer prediction.

– The length of the colored bars corresponds to the strength of each feature's impact on the final decision.

Figure 3.10: LIME Interpretation Result Interface

At the bottom of the results page (Figure 3.11), This section includes a ranked list of the features that most influenced the model's decision, along with a visual guide explaining how to interpret these results. It also provides color-coded indicators, value types, and a general interpretation.

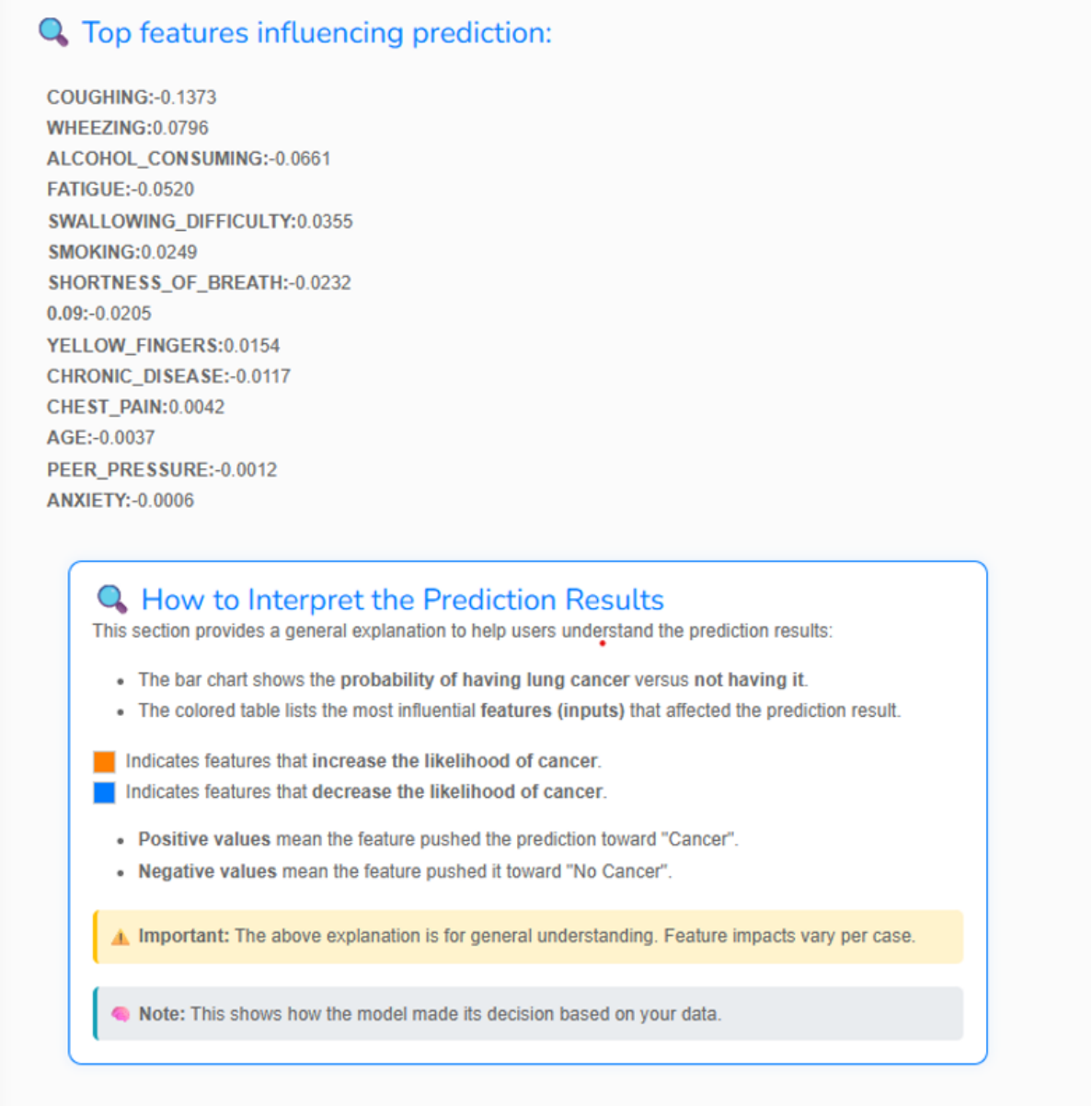

Figure 3.11: Explanation of Prediction Results and Feature Impact

4.SHAP Global Interpretation Interface

In this section, we present the **SHAP Global Interpretation** interface of our application.

- This interface displays four visual components designed to help interpret the global behavior of a deep learning model using SHAP values.
- The top left component shows the individual impact of features on the model's output for a specific instance.
- The bottom left component presents how each individual feature influenced the prediction outcome for each sample in the dataset.

- The top right component lists the features along with their respective importance scores based on SHAP values.
- The bottom right component illustrates how each feature contributes to the model prediction across all data points, showing the distribution and influence direction.

Additionally, a blue interpretation guide at the bottom provides brief instructions on how to read and understand the displayed visualizations.

The left panel also includes navigation buttons such as Tasks, Explainable AI, Result, and FAQ .

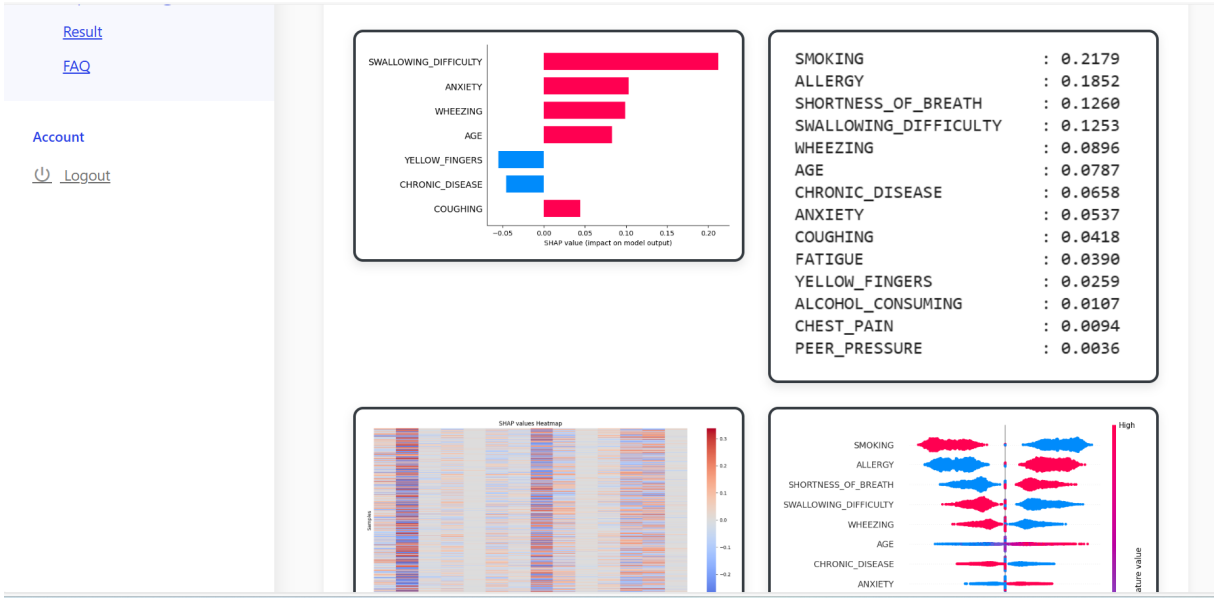

Figure 3.12: SHAP-Based Model Insight Interface

5.Frequently Asked Questions Interface

In this section, we present the Frequently Asked Questions (FAQ) interface of our application. This interface is designed to help users understand common insights extracted from the dataset by automatically answering predefined questions.

- The main panel titled “Frequently Asked Questions” displays multiple question-answer pairs related to lung cancer occurrences based on different symptoms or behaviors.
- Each entry shows a specific condition (e.g., coughing, alcohol consumption, fatigue, smoking) and the number of individuals who had lung cancer out of the total in the dataset.
- The answers are presented in a simple, readable format to help users quickly interpret the statistics without requiring advanced data analysis skills.

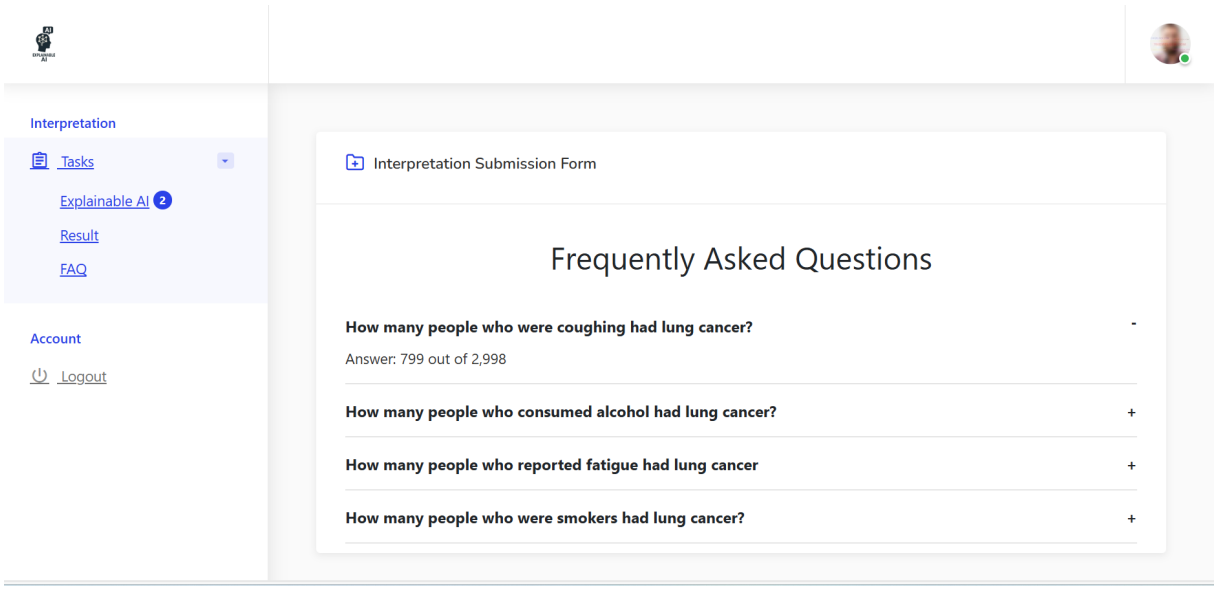

Figure 3.13: Interactive Medical FAQ System Interface

6. Knowledge Graph Interface for Patient Medical Data

In this section (Figure 3.14), we present the Knowledge Graph interface of our application. This interface displays the knowledge graph of 20 patients in our dataset. The graph represents entities such as patients, symptoms, diseases, and risk factors, along with the relationships between them. It enables the visualization of how patients in the dataset are interconnected through shared symptoms and risk factors, which helps in medical reasoning and model explanation.

- At the center of the graph, we find a highly connected node, which is typically a patient (e.g., `patient_4`) or a central medical concept such as Lung Cancer. This central node acts as a hub, connecting various symptoms (e.g., Coughing, Wheezing), risk factors (e.g., Smoking, Alcohol Consuming), and diagnoses.
- Green nodes represent patients, yellow nodes indicate symptoms, and red nodes denote risk factors.
- The arrows between nodes illustrate relationships, such as:
  - `hasSymptom`: linking a patient to a symptom.
  - `hasFactor`: linking a patient to a risk factor.
  - `DIAGNOSED_WITH`: connecting a patient to a diagnosis.

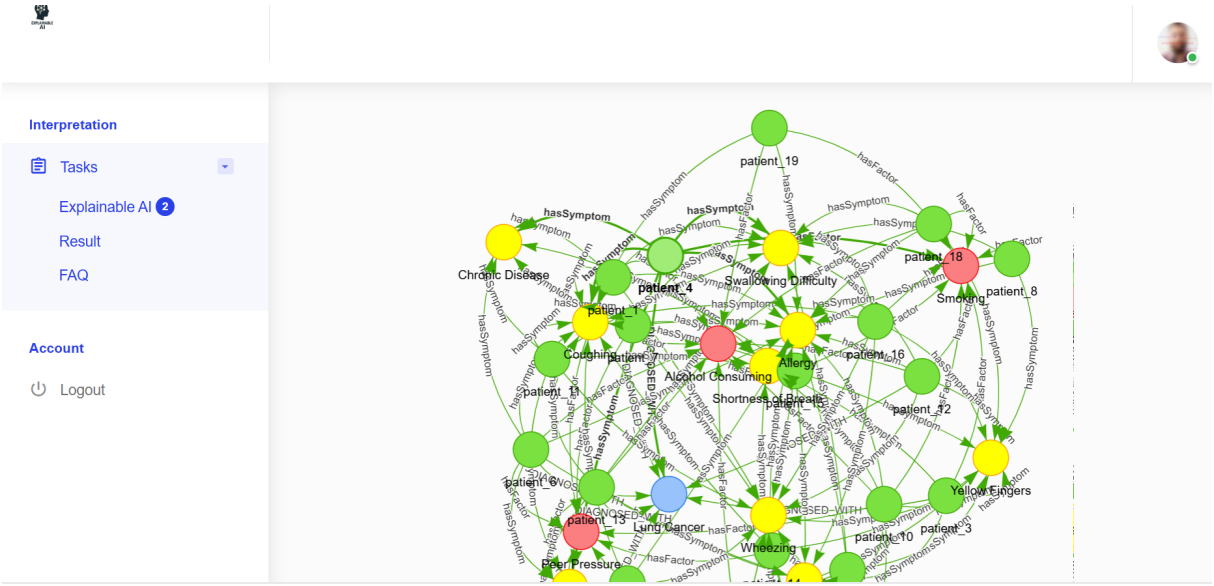

Figure 3.14: Medical Knowledge Graph Visualization of Patients and Symptoms

3.9 Conclusion

In this chapter, we described the tools and environment used for the development of our system. Then we presented the testing procedure used to evaluate the system, followed by an interpretation of the results obtained and the exploitation of the necessary components. Finally, we presented the user interfaces developed for the application.

General Conclusion

The medical field is one of the most sensitive and vital domains in human life, as the quality of diagnostic and therapeutic decisions has a direct impact on patient's health and well-being. With the continuous growth in the volume and complexity of medical data, there is an urgent need to develop intelligent tools capable of analyzing this information in an efficient, accurate, and reliable manner.

In this context, the Knowledge Graph emerges as an effective tool for organizing medical knowledge in a flexible and interconnected way, enhancing the capabilities of intelligent models in the field of diagnosis. However, the adoption of artificial intelligence in healthcare remains highly dependent on user's trust—trust that cannot be established without transparency in how these models operate. This highlights the importance of Explainable AI, which aims to produce models whose decisions can be interpreted, allowing physicians to understand the reasoning behind each decision and adopt these systems confidently and effectively.

In our project, we focused on designing the proposed medical diagnostic system. We began by selecting a specialized lung cancer dataset, given the importance and prevalence of this disease. The data was then prepared and processed. Subsequently, we constructed a knowledge graph to semantically organize the information.

We extracted graph embeddings to serve as inputs to the learning model. Then, we developed a diagnostic system based on the GAT model to predict medical diagnosis. Finally, explainable AI techniques were applied to the developed model to provide understandable explanations of the diagnostic results, in both local and global level.

In implementation, we focused on the practical and technical aspects related to the implementation and evaluation of the proposed system. We began by describing the working environment and the software tools used. We then presented the adopted testing methodology for system evaluation, followed by an analysis and interpretation of the

obtained results. We also addressed how the necessary components of the system were integrated, and concluded by presenting the user interfaces developed to facilitate interaction with the system for end users.

As for the future perspectives of this work, we plan to further develop the system by adding new functionalities such as suggesting appropriate treatments for medical conditions. We also aim to enhance the current dataset by incorporating a larger number of medical cases and including chest X-ray images to enrich the model. Additionally, we intend to experiment with new graph embedding algorithms. Lastly, one of the most promising directions we aim to explore is the application of our system in other medical domains by utilizing different knowledge bases. Due to its transfer learning capabilities, the system can be adapted and reused in various contexts, making it a strong candidate for broader deployment.

Bibliography

[1] Chen, Z., Wang, Y., Zhao, B., Cheng, J., Zhao, X., & Duan, Z. (2020). Knowledge graph completion: A review. *IEEE Access, 8*, 192435–192456. https://doi.org/10.1109/ACCESS.2020.3030076

[2] Kejriwal, M. (2019). *Domain-Specific Knowledge Graph Construction* (pp. 2–3). Springer Cham. https://doi.org/10.1007/978-3-030-12375-8

[3] Blumauer, A., & Nagy, H. (2020). *The Knowledge Graph Cookbook* (pp. 19–22). edition mono/monochrom.

[4] Sowa, J. F. (1976). Conceptual graphs for a data base interface. *IBM Journal of Research and Development*, *20*(4), 336–357. http://www.jfsowa.com/pubs/cg1976.pdf

[5] Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The Semantic Web. *Scientific American*, *284*(5), 34–43. https://www.scientificamerican.com/article/the-semantic-web/

[6] The GQL Manifesto. *One Property Graph Query Language*. from https://gql.today/

[7] Zaidi, E., & De Simoni, G. (2019). *Augmented data catalogs: Now an enterprise must-have for data and analytics leaders*. Gartner, Inc. https://www.gartner.com/en/documents/3957301

[8] Kejriwal, M. (2019). *Domain-Specific Knowledge Graph Construction* (p. 5). Springer Cham. https://doi.org/10.1007/978-3-030-12375-8

[9] Blumauer, A., & Nagy, H. (2020). *The Knowledge Graph Cookbook* (pp. 22–23). edition mono/monochrom.

[10] Yu, G., Tabatabaei, M., Mezei, J., Zhong, Q., Chen, S., Li, Z., Li, J., Shu, L., & Shu, Q. (2022). Improving chronic disease management for children with knowledge graphs and artificial intelligence. *Expert Systems with Applications, 201*, 117026. https://doi.org/10.1016/j.eswa.2022.117026

[11] Najjar, R. (2023). Redefining radiology: A review of artificial intelligence integration in medical imaging. *Diagnostics, 13*(27), 2760. https://doi.org/10.3390/diagnostics13172760

[12] Chai, X. (2020). Diagnosis method of thyroid disease combining knowledge graph and deep learning. *IEEE Access, 9*, 123456–123467. https://doi.org/10.1109/ACCESS.2020.3016676

[13] Tan, J., Jin, Y., & Lamy, J. B. (2021). Explainable artificial intelligence in clinical decision support systems: Opportunities and challenges. *Applied Sciences, 11*(15), 5088. https://doi.org/10.3390/app11115088

[14] Kim, S. Y., Ryu, O. J., Kim, D. H., Kim, M. J., & Kim, H. J. (2024). A novel explainable AI-based clinical decision support system framework for medical applications. *Applied Sciences, 14*, 6638.

[15] Xie, Y., Zhang, Y., Wang, Y., & Liu, Y. (2024). GRAPHCARE: Enhancing healthcare predictions with personalized knowledge graphs. In *Proceedings of the International Conference on Learning Representations (ICLR).*

[16] Seo, S., Cheon, H., Kim, H., & Hyun, D. (2023). Structural quality metrics to evaluate knowledge graph quality. *Semantic Web*, *14*(1), 5-6. https://doi.org/10.3233/SW-223366

[17] Paulheim, H. (2017). Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic Web, 8*(3). https://doi.org/10.3233/SW-160218

[18] Zaveri, A., Rula, A., Maurino, A., Pietrobon, R., Lehmann, J., & Auer, S. (2016). Quality assessment for linked data: A survey. *Semantic Web, 7*(1), 63–93. https://doi.org/10.3233/SW-150175

[19] Agrawal, G., Deng, Y., Park, J., Liu, H., & Chen, Y.-C. (2022). Building knowledge graphs from unstructured texts: Applications and impact analyses in cybersecurity education. *Information, 13*, 526. https://doi.org/10.3390/info13110526

[20] Fensel, D., Simsek, U., Angele, K., Huaman, E., Kärle, E., Panasiuk, O., Toma, I., Umbrich, J., & Wahler, A. (2023). Knowledge graphs—Methodology, tools and selected use cases. *SN Computer Science, 4*, 16. https://doi.org/10.1145/3477314.3507655

[21] Neo4j. (2025). Neo4j graph database platform. Retrieved June 2, 2025, from https://neo4j.com

[22] Amazon Web Services. (2025). Amazon Neptune – Graph database service. Retrieved June 2, 2025, from https://aws.amazon.com/neptune/

[23] Stanford Center for Biomedical Informatics Research. (2025). Protégé ontology editor and knowledge acquisition system. Retrieved June 2, 2025, from https://protege.stanford.edu

[24] Adadi, A., & Berrada, M. (2018). Peeking inside the black-box: A survey on explainable artificial intelligence (XAI). *IEEE Access, 6*, 52138–52160. https://doi.org/10.1109/ACCESS.2018.2870052

[25] Doshi-Velez, F., & Kim, B. (2017). Towards a rigorous science of interpretable machine learning. https://arxiv.org/abs/1702.08608

[26] Holzinger, A., Biemann, C., Pattichis, C. S., & Kell, D. B. (2017). What do we need to build explainable AI systems for the medical domain? https://arxiv.org/abs/1712.09923

[27] Tonekaboni, S., Joshi, S., McCradden, M. D., & Goldenberg, A. (2019). What clinicians want: Contextualizing explainable machine learning for clinical end use. In *Proceedings of the Machine Learning for Healthcare Conference (MLHC)*. PMLR.

[28] Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). "Why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM. https://doi.org/10.1145/2939672.2939778

[29] Rajkomar, A., Dean, J., & Kohane, I. (2019). Machine learning in medicine. *New England Journal of Medicine, 380*(14). https://doi.org/10.1056/NEJMra1814259

[30] Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). "Why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM. https://doi.org/10.1145/2939672.2939778

[31] Lundberg, S. M., & Lee, S.-I. (2017). A unified approach to interpreting model predictions. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NeurIPS 2017)*. Curran Associates, Inc. https://papers.nips.cc/paper_files/paper/2017/hash/8a20a8621978632d76c43dfd28b67767-Abstract.html

[32] Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *Annals of Statistics, 29*(5). https://doi.org/10.1214/aos/1013203451

[33] Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning, 1*(1), 81–106. https://doi.org/10.1007/BF00116251

[34] Hosmer, D. W., & Lemeshow, S. (2000). *Applied logistic regression* (2nd ed.). Wiley.

[35] Tonekaboni, S., Joshi, S., McCradden, M. D., & Goldenberg, A. (2019). What clinicians want: Contextualizing explainable machine learning for clinical end use. In *Proceedings of the Conference on Health, Inference, and Learning (CHIL 2019)*. ACM. https://doi.org/10.1145/3368555.3384464

[36] Lundberg, S. M., Erion, G., Chen, H., DeGrave, A., Prutkin, J. M., Nair, B., ... & Lee, S.-I. (2018). Explainable machine-learning predictions for the Analysis of Complex Datasets in Healthcare. *Nature Biomedical Engineering, 2*. https://doi.org/10.1038/s41551-018-0304-0

[37] Holzinger, A., Biemann, C., Pattichis, C. S., & Kell, D. B. (2017). What do we need to build explainable AI systems for the medical domain?. https://arxiv.org/abs/1712.09923

[38] Shortliffe, E. H. (1976). *Computer-based medical consultations: MYCIN*. Elsevier.

[39] Vailati-Riboni, M., Palombo, V., & Loor, J. J. (2017). What are omics sciences? In B. N. Ametaj (Ed.), *Periparturient diseases of dairy cows: A systems biology approach* (pp. 1–7). Springer International Publishing.

[40] Maloy, C. (2022). *Library Guides: Data Resources in the Health Sciences: Clinical Data*. Retrieved from https://guides.lib.uw.edu/hsl/data/findclin.

[41] Sow, D., Turaga, D. S., & Schmidt, M. (2013). Mining of sensor data in healthcare: A survey. In C. C. Aggarwal (Ed.), *Managing and mining sensor data* (pp. 459–504). Springer: Boston, MA, USA.

[42] Wang, H., Huang, H., Ding, C., & Nie, F. (2012). Predicting protein-protein interactions from multimodal biological data sources via nonnegative matrix tri-factorization. In B. Chor (Ed.), *Proceedings of Research in Computational Molecular Biology* (pp. 314–325). Springer: Berlin/Heidelberg, Germany.

[43] Wu, Q., Wang, Z., Li, C., Ye, Y., Li, Y., & Sun, N. (2015). Protein functional properties prediction in sparsely-label PPI networks through regularized non-negative matrix factorization. *BMC Systems Biology*, 9(Suppl 9), S9.

[44] Alshahrani, M., Khan, M. A., Maddouri, O., Kinjo, A. R., Queralt-Rosinach, N., & Hoehndorf, R. (2017). Neuro-symbolic representation learning on biological knowledge graphs. *Bioinformatics*, 33(17), 2723–2730.

[45] Trivodaliev, K., Josifoski, M., & Kalajdziski, S. (2018). Deep learning the protein function in protein interaction networks. In S. Kalajdziski & N. Ackovska (Eds.), *ICT Innovations 2018. Engineering and Life Sciences* (pp. 185–197). Springer International Publishing: Cham, Switzerland.

[46] Wu, F. X., Shen, Z., Zhang, Y. H., Han, K., Nandi, A. K., Honig, B., & Huang, D. S. (2017). miRNA-disease association prediction with collaborative matrix factorization. *Complexity*, 2017, Article ID 2498957.

[47] Yang, K., Wang, N., Liu, G., Wang, R., Yu, J., Zhang, R., Chen, J., & Zhou, X. (2018). Heterogeneous network embedding for identifying symptom candidate genes. *Journal of the American Medical Informatics Association*, 25(11), 1452–1459.

[48] Xu, B., Liu, Y., Yu, S., Wang, L., Dong, J., Lin, H., Yang, Z., Wang, J., & Xia, F. (2019). A network embedding model for pathogenic genes prediction by multi-path random walking on heterogeneous network. *BMC Medical Genomics*, 12, 188.

[49] Wang, X., Gong, Y., Yi, J., & Zhang, W. (2019). Predicting gene-disease associations from the heterogeneous network using graph embedding. In *Proceedings of the IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, San Diego, CA, USA, November 18–21, 2019 (pp. 504–511).

[50] Wang, M., Liu, M., Liu, J., Wang, S., Long, G., & Qian, B. (2017). Safe medicine recommendation via medical knowledge graph embedding.

[51] Zhao, C., Jiang, J., Guan, Y., Guo, X., & Hen, B. (2018). EMR-based medical knowledge representation and inference via Markov random fields and distributed representation learning. *Artificial Intelligence in Medicine*, 87, 49–59.

[52] Choi, E., Bahadori, M. T., Song, L., Stewart, W. F., & Sun, J. (2017). GRAM: Graph-based attention model for healthcare representation learning. In *Proceedings of the International Conference on Knowledge Discovery Data Mining*, Halifax, NS, Canada, August 13–17, 2017 (pp. 787–795).

[53] Wang, S., Chang, X., Li, X., Long, G., Yao, L., Sheng, Q. Z. (2016). Diagnosis code assignment using sparsity-based disease correlation embedding. *IEEE Transactions on Knowledge and Data Engineering*, 28(12), 3191–3202. https://doi.org/10.1109/TKDE.2016.2605687

[54] Wang, M., Ma, X., Si, J., Tang, H., Wang, H., Li, T., Ouyang, W., Gong, L., Tang, Y., He, X., & others. (2021). Adverse drug reaction discovery using a tumor-biomarker knowledge graph. *Frontiers in Genetics*, 11, 1737.

[55] World Health Organization. (2023). Lung cancer. *WHO Fact Sheet*. Retrieved from https://www.who.int/news-room/fact-sheets/detail/lung-cancer

[56] Nath, A. (2022). Lung cancer dataset [Data set]. Kaggle. https://www.kaggle.com/datasets/akashnath29/lung-cancer-dataset