Grenze International Journal of Engineering and Technology, June Issue

MediGraph: Semantic Knowledge Graph for Healthcare

Qudsiya Naaz[1], Hasnain Raza Khan[2], Mirza Rehan Beg[3], Hasib Ur Rahman[4], Alkama Ansari[5] and Ayush Chaudhary[6]
[1]Professor, Dept.of Computer Science & Engineering Anjuman College of Engineering & Technology Nagpur, India
[2-6]B. Tech Undergraduate Students, Dept. of Computer Science & Engineering Anjuman College of Engineering & Technology, Nagpur, India
Email: thehrkofficial@gmail.com, rm688145@gmail.com

***Abstract*— A portion of medical data is available only in written form, such as medical journals, reports, and online health articles. It can be seen that the unstructured written data is complicated and cannot be rearranged and reused in its current form for analysis. Most available solutions deal with only the extraction of medical terminology, or over-rely on some form of empirical or heuristic rule with deep domain knowledge that limits flexibility or practical application. Considering this, the focus of this project is to provide an uncomplicated and practical means to convert healthcare-related unstructured text to knowledge graphs. The system was developed in Python and uses basic natural language processing. It offers a linear and sequential workflow approach where unstructured raw text data is first, cleaned and prepared. Following this, core medical entities are recognized in the processed text, such as those dealing with specific symptoms, diseases, and treatments. Once the recognition of entities is done, contextually appropriate and co-occurring meaningful relationships are created. Using the Neo4j database, relationships or graphs are stored so that users may confirm or verify them visually. Even though some additional pre-processing is still required for the system, it was able to provide reliable results for many common medical terminologies during the testing phase.**

***Index Terms*— Knowledge Graph, Healthcare NLP, Entity Extraction, Triple Generation, Neo4j, Data Preprocessing.**

I. INTRODUCTION

Knowledge Graphs A knowledge graph organizes information in a way that connects related entities. Instead of just tables of data, the graph shows data entities as points, and connections show relationships between items. This structure emphasizes a relationship more than the individual entities. A graph makes it easier to see the relationship between data items (nodes). Unstructured Data in Healthcare The healthcare industry contains a lot of information that is in text form, which is known as unstructured data. Doctor's notes, medical reports, discharge summaries, and written symptom descriptions are examples. Unstructured healthcare data contains a lot of information that is valuable, but it is difficult to access because it is contained within long-form text. The text contains unarticulated relationships across the documents to describe the linkages between symptoms, diseases, and treatments.

Challenges in Existing Systems: The major databases in use today are unable to fully harness text data in healthcare. Fundamental databases are effective with structured data but paralyze with free text. Some databases attempt to capture only some of the medical terminology, but there is no explanation of how the terms are interrelated.

Grenze ID: 01.GIJET.12.2.70_22
© Grenze Scientific Society, 2026

Other approaches rely on manually constructed rules or medical ontologies, both of which are incredibly time consuming and require extensive domain expertise. This is why many systems remain extremely difficult to scale, and are non-adaptive to new data.

Motivation and Contribution: The primary objective of this project is to enhance comprehension of healthcare text by transmuting it into a more organized framework. This research is centred around the creation of a structured medical knowledge graph. The architecture of the system is broken down into discrete blocks so that each component can be upgraded in the future. In contrast to most previous systems, this framework not only captures medical terminology, but also establishes relationships among terms, which are then processed in a graph database. This provides a competitive edge to the system in the context of the forthcoming healthcare applications like the analysis of medical data and the support of decision making.

II. Literature Survey

There has been an exponential increase in the studies focused on Knowledge Graphs and NLP. Many studies focus on the extraction of entities, detection of relations, and the construction of graphs. This sub-section summarizes some relevant studies and defines some of their shortcomings in relation to the proposed system in this paper. Extraction of Medical Entities Using Rule-Based NLP Systems Method: Rule-based patterns and medical lexicons [1]. Data: symptoms descriptions and medical notes and dataset. Result: Good for common medical terms. Limitation: For unseen or complex sentences, it gives poor accuracy. Improvement in Our Work: Uses NLP preprocessing and flexible extraction workflows that work on various health texts. Ontology-Based Healthcare Knowledge Modeling Method: Manual ontology creation and labelling [2]. Data: Small curated medical documents. Result: Well-structured relations. Limitation: Heavy manual effort, not scalable. Improvement: Our system reduces dependency on manual rules by automating extraction and relation or triples creation. Deep Learning Models for Clinical Named Entity Recognition Method: Bi-LSTM and CNN-based NER models [3]. Data: Labeled biomedical datasets. Result: Good accuracy when training data is large. Limitation: Requires huge datasets and high training cost. Improvement: Our lightweight NLP approach works with smaller datasets and still produces clean entity sets. Transformer-Based Healthcare Text Understanding Method:BioBERT/BERT for extracting medical entities [4]. Data: Large biomedical corpora. Result: High performance for entity detection. Limitation: Slow, hardwareheavy, difficult to integrate with graphDBs.Improvement: Our pipeline is simple, faster, and directly integrated with Neo4j triple creation. Knowledge Graphs for Disease–Symptom Mapping Method: Manual mapping of predefined disease– symptom relations [5]. Data: Disease–symptom dataset from health websites. Result: Simple relational graph. Limitation: Relations are static; cannot expand automatically. Improvement: Our system dynamically generates relationships using extracted triples. Neo4j-Based Biomedical Graph Visualization Method: Creation of Manual node and edge in Neo4j [6]. Data:Small biomedical knowledge base. Result: Good visualization and fast querying. Limitation: Only storage; no NLP pipeline before graph creation. Improvement: Our model automates preprocessing → extraction → triple → Neo4j storage. Healthcare Recommendation Using Graph Networks Method: Graph network models for recommending treatments [7]. Data: Patient symptoms and treatment logs. Result: Simple and useful recommendations. Limitation: limited coverage and small datasets. Improvement: Our graph expands automatically and supports large-scale symptoms–diseases–treatments. Medical Symptom Analysis from Mining of Text Method: Tokenization, TF-IDF, and simple NLP techniques [8]. Data: Online health portals symptom descriptions. Outcome: Basic symptom descriptions clustering. Limitation: No relational understanding, disease links cannot be interpret. Improvement: Our knowledge graph captures relations between symptoms, diseases, and treatments. Graph Construction from Wikipedia Medical Articles Method: Relation extraction from semi-structured Wikipedia medical texts [9]. Data: Wiki-based medical pages. Result: Good general knowledge graph. Limitation: Not domain-accurate for real healthcare use. Improvement: Our dataset focuses on healthcare terms and produces more domain-correct triples. NLP Pipelines in Clinical Decision Support Systems Method: NLP pipelines to convert unstructured EHR data into structured form [10]. Data: Electronic health records (EHR). Result: Improves organization of data. Limitation: Pipelines are large, complex, and not opensource. Improvement: Our system can be easily adapted for various medical sub-domains as the system is simple, modular, lightweight.

III. Key Contributions of the Proposed Framework

This project focuses on solving a practical problem found in healthcare data, where most information is written in free text and is hard to organize. The main idea behind this work is to convert such unstructured healthcare

text into a knowledge graph in a simple and automatic way. Many earlier studies concentrate only on one task, such as finding entities or defining rules for relations. In this regard, we attempt to integrate all the crucial components of the system to create a single pipeline with the least amount of human intervention. The modular nature of this project is, arguably, its most significant advantage. The system is designed with stages: pipeline construction, entity recognition, relations, and graphs. Because of this separation, the system as a whole is more comprehensible and more straightforward to manipulate. In the case of future improvements, separate modules can be upgraded with little or no changes to the whole system and its operations. This is an advantage most other systems are lacking. Another key merit lies in the technique employed in relationship creation. The system establishes direct and relevant relationships among the selected healthcare entities (symptoms, diseases, and treatments) without depending on large datasets, complex transformer training, or preset ontologies of domains. This technique minimizes the need for manual work and preserves the relevance to the domain. Typically, and even in the case of medical data, the general-purpose knowledge graph systems deployed and underperformed. However, focusing solely on healthcare concepts (symptoms, diseases, and treatments), the system yields worthwhile results. The data obtained are stored in Neo4j, which enables clear relationship visualization. This integrates natural language processing and graph storage, which many previous works fail to do.

On the balance, the work lays a good foundation for further development. The current system successfully manages preprocessing, data extraction, and the creation of the graph. Additional and novel features such as enhanced relationships or medical reasoning will be possible. The set of differentiating elements in this project is its modular design, simplicity, lesser reliance on transformers and ontologies, healthcare focus, and graph visualization.

IV. Proposed System

Proposed system represents how we convert scattered and unstructured healthcare data into a well-structured knowledge graph. It uses a modern approach, involving step-by-step development and easy future addition. The system consists of the following main modules, as shown in Figure 1: Collection of Data Healthcare-related information such as symptoms, diseases and treatments including the descriptions of each are collected from various resources which is in the form of unstructured healthcare data. This raw data show lacks a predefined structure assist as an input data [1]. Data Preprocessing (Module-1): Preprocessing steps perform certain operation on collected data such as noise removal and tokenization to improve the quality data. These steps remove Redundancy and Noise make data in a Normalized and required format [2]. Entity Extraction: After preprocessing, to identify important and main core features and entities such as symptoms, diseases and treatments, Several NLP techniques are applied. These selected entities show medical concepts and act as a node on knowledge graph. Triple Generation: The Known entities are converted into Subject–Relation– Object triples. For example, a relationship such as ―Fever → indicates → Dengue‖ represents a meaningful medical association. These triples define the strong relation between nodes and form the edges of the knowledge graph. Graph Storage and Visualization Using Neo4j: Information which are extracted and make relationships by connecting extracted data are stored in Neo4j graph database that represents clear visualized graph consist nodes allow traversal and future expansion of healthcare knowledge graph based on data.[4]. Query System: The system is designed in a way that user can make a query from Frontend to Backend. Users will be able to retrieve meaningful medical information, such as finding possible diseases and exploring treatment options based on User query. This module will enhance the practical usability of the system in real-world healthcare applications.

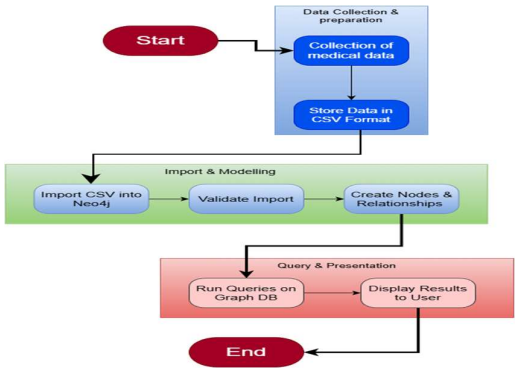

Figure 1: Process Flow Diagram of Entity Extraction and Graph Creation

V. IMPLEMENTATION

This section represents how the proposed system implemented a healthcare knowledge graph from unstructured data. The implementation mainly pays attention on Module-1, which includes preprocessing of data, Extraction and Identification of main entity, relation generation, and creation of graph. The complete system is developed using Python along with NLP libraries and the Neo4j graph database. The well-manner structure helps to expand this system to scale and expand in future stages. Data Preprocessing The main reason to apply Preprocessing technique as the system takes raw, scattered and shapeless healthcare-related text as input form. This text includes information about symptoms, diseases, and treatments gathered from various sources such as medical articles, online health websites, and reports. Since the raw data is unstructured, preprocessing is required before implementing any NLP techniques. Following are the steps of Preprocessing: Cleaning of Text: Removal of unnecessary characters, symbols HTML characters to reduce noise. Normalization: For the Standardization and Consistency of medical terms, Normalization is used to reduce Redundancy, and the text is converted to lowercase. Tokenization: For further analysis, Individual words and tokens are created by splitting the Sentences. Stop word Removal : Unnecessary word such as "is", “and” "the" is removed when required because it does not add meaningful information. Sentence Segmentation: To maintain the contextual meaning text are separated into sentences. These steps clear that the input text is clean, consistent, and reliable for correct entity extraction. Entity Extraction In Extraction phase, To identify important healthcare-related entities from the data and Information, NLP Techniques are used. Named Entity Recognition (NER) is performed using the spaCy library with healthcare-oriented rules. The entities such as symptoms, diseases, and treatments from each system automatically detect by the system using NLP. Example: Input Sentence: Common symptoms such as Fever and Headache represent disease Dengue. Extracted Entities: Fever (Symptom), Headache (Symptom), Dengue (Disease). In the healthcare knowledge graph extracted entities act as the nodes and core components. Triple Generation After the identification of entities, meaningful relationships are formed between entities. These relationships are represented in the form of Subject–Relation–Object (S–R– O) triples relation. Each triple consists a basic medical interconnection derived from the text. Below are the examples of Relationship created by the system: Fever → indicates → Dengue , Headache → interconnected_with → Dengue , Paracetamol → treats → Fever The semantic structure required to make a knowledge graph provided by these related triplets. Graph Creation in Neo4j The Neo4j graph database are used to stored generated triples .That generated triples consist of entity which is represented as a node and the relationships between them are represented as edges. For automatically creation of nodes and relationships in the Neo4j graph database, Cypher queries are executed with the help of python. To inspect visually and confirm the graph structure Neo4j browser is used. Cypher Query example: CREATE (s:Symptom {name:'Fever'})-[:INDICATES]- >(d:Disease {name:'Dengue'}) The relationships between symptoms, diseases, and treatments, clearly shows the graph based representation ,makes the data easy to understand and query. Terminal Output At the time of execution, the system prints intermediate outputs in the terminal, including: Preprocessed text Extracted entities Generated triples Status messages for node and relationship creation in Neo4j To verify that each stage of Module-1 is functioning correctly and provide immediate feedback at the time execution, Output on terminal and graph creates on Neo4j database are helpful.

VI. RESULT

Initial Results of Module-1 Implementation are shown in Table I.

TABLE I: ANALYSIS OF TERMINAL OUTPUT AND GRAPH VISUALIZATION

Aspects	Descriptions
Input Data	Raw and Scattered data
Preprocessing Text	clean and normalize text for NLP tasks
Entity Extraction	Identification of Important Entities
Triple Generation	Creation of Relation of entity and triples automatically.
Backend Output	Triples Entities print on the Terminal as an output.
Graph Creation	Nodes and relationships generated in Neo4j Database.
Visualization	Connection of Nodes such as symptoms, diseases and treatment or healthcare graph visible in Neo4j Database
Execution Status	Successful end-to-end Module-1 workflow

The results obtained show that the developed system can convert unstructured healthcare text into a well-organized knowledge representation. After the preprocessing stage, the system correctly identifies important medical information such as symptoms, diseases, and treatments from the input text. These extracted entities are

then used to form Subject– Relation–Object triples. The generated triples are printed in the terminal, allowing verification of the internal processing at each step. The same triples output has also appeared as nodes and relationships which are created and stored into the Neo4j graph database. The graph clearly shows meaningful connections between symptoms, diseases, and treatments When viewed in the Neo4j Browser. This visual output verified that the relationships created from the data are logically represented in the graph format is correct. The results clarify that Module-1 performs all its intended tasks, which includes processing of text, identification of entity from data, formation of nodes and relation, and creation of visual graph. The execution of each stage of system demonstrates that the proposed approach is practical and suitable for handling health care related data. Finally, A complete and automated route was established by Module-1 e that transform raw and unstructured medical data into a structured and well-mannered knowledge graph. The modular structure combined with the focus on the specific needs of the healthcare sector ensures both reliability and adjustability of the implementation. This module also incorporates advanced healthcare applications, deep semantic analysis, and improved relation extraction.

VII. Conclusion

This document details the primary module of the healthcare knowledge graph system which converts unstructured healthcare text, including clinical notes, descriptions of symptoms, and articles, into a knowledge graph. Module-1 has successfully completed the preprocessing of raw text, the building of Subject-Relation-Object triples and their storage into the Neo4j system, and the extraction of symptoms, diseases, and treatments. The system also demonstrates the preprocessing of text, the extraction of critical medical entities, the generation of a relation based on the extracted entities, and the status of the insertion into Neo4j, thus confirming the workflow of the system.

The Neo4j graph produced by the system shows the healthcare entities that have been created as nodes in the Neo4j graph database, which provides a clear representation of the interconnected healthcare data. The project captures the principles of automation, modularity, and scalability, thus laying a semantic foundation for the future modules.

Future Scope

Extraction and interpretation of relations: Assistance with complex and/or latent relationships will be useful. Based on symptoms and disease, doctors will be able to provide recommendations for treatment and diagnosis with ease.

Integration with larger datasets: In order to create more precise graphs, large datasets of medical data will be needed. User personalization: Recommendations can be personalized based on user medical history, user location with disease, and the preferences of the user.

Real World Usage

Knowledge graphs assist doctors in identifying potential illnesses based on provided symptoms. Knowledge graphs can provide possible treatments for common or uncommon illnesses. It can assist in medical research by revealing the connections between symptoms, diseases, and treatments. It offers a quick and structured method to work with large amounts of unstructured healthcare data.

References

[1] T. Navigli and S. P. Ponzetto, ―BabelNet: The Automatic Construction, Evaluation and Application of a Wide-Coverage Multilingual Semantic Network,‖ Artificial Intelligence, vol. 193, pp. 217–250, 2012.

[2] M. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives, ―DBpedia: A Nucleus for a Web of Open Data,‖ The Semantic Web, pp. 722–735, 2007.

[3] F. Li, X. Zhang, and Y. Liu, ―Text Preprocessing Techniques for NLP: A Comprehensive Survey,‖ Journal of Computational Science, vol. 45, pp. 101–115, 2020.

[4] R. Fricke, A. R. Silva, and F. Z. de Carvalho, ―Neo4j Applications in Biomedical Knowledge Graphs,‖ Procedia Computer Science, vol. 144, pp. 214–221, 2018.

[5] Y. Kim, ―Convolutional Neural Networks for Sentence Classification,‖ EMNLP, 2014.

[6] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, ―BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,‖ NAACL-HLT, 2019.

[7] R. Navigli, ―Word Sense Disambiguation: A Survey,‖ ACM Computing Surveys, vol. 41, no. 2, pp. 1–69, 2009.

[8] P. Shvaiko and J. Euzenat, ―Ontology Matching: State of the Art Survey,‖ Journal on Data Semantics, vol. 3, no. 1, pp. 1–25, 2005.
[9] H. Chen, Y. Zhang, and D. Yu, ―Building Healthcare Knowledge Graphs with NLP and Deep Learning,‖ Journal of Biomedical Informatics, vol. 98, 2019.
[10] L. Wang, S. Li, and Z. Xu, ―Entity and Relation Extraction from Biomedical Text Using Deep Learning,‖ BMC Bioinformatics, vol. 21, 2020