A Rule-Based Expert System for Medical Diagnosis: Utilizing logic programming and knowledge representation techniques

Fatema Afsan Ema (2022281642), Rakibul Hasan Ridoy (1731339042),
Md Mashikur Rahaman (2011033642), Md Mehidi Hasan Khan (2013957642),
Nafsi Omar Prokrety (2022326642)
Department of Electrical and Computer Engineering
North South University
CSE 440 Group 03

***Abstract*—This report details the design and ongoing development of a prototype Medical Diagnosis Expert System. The system is engineered to assist healthcare professionals by diagnosing diseases based on patient-reported symptoms. Leveraging fundamental principles of artificial intelligence, the system employs a rule-based knowledge representation and a forward chaining inference engine to derive logical conclusions from a set of initial facts. The core architecture is composed of a distinct Knowledge Base (KB) and an Inference Engine, which ensures modularity and scalability. The current implementation, developed in Python, successfully diagnoses a predefined set of illnesses based on user input via a command-line interface. This report outlines the system's architecture, implementation details, preliminary results, and the strategic direction for future enhancements, which include incorporating uncertainty management and developing a more sophisticated user interface.**

***Index Terms*—Expert System, Medical Diagnosis, Artificial Intelligence, Knowledge Representation, Rule-Based System.**

I. Introduction

In modern healthcare, accurate and timely diagnosis is crucial for effective treatment and improved patient outcomes. With the increasing complexity of diseases, the sheer volume of medical knowledge, and the need for constant updates, healthcare professionals can face challenges in making informed decisions. In this context, expert systems—a branch of artificial intelligence (AI)—have become a powerful tool for assisting healthcare providers by mimicking the diagnostic reasoning of human experts.

Expert systems, a prominent branch of applied Artificial Intelligence (AI), are designed to emulate the decision-making ability of a human expert within a specific domain [1]. In the medical field, they serve as powerful tools for diagnosis, treatment planning, and consultation, processing vast amounts of information to provide reasoned conclusions [2].

The Medical Diagnosis Expert System aims to create a decision-support system that helps healthcare professionals diagnose diseases based on patient symptoms, medical history, and other clinical data. By using logic programming and knowledge representation techniques, this system can infer likely diagnoses and provide recommendations for further tests or treatment plans.

This project follows a structured plan, beginning with the implementation of a core system featuring a Knowledge Base (KB) populated with medical rules and an Inference Engine that applies these rules. The current system utilizes a forward chaining algorithm, a data-driven approach that starts with known facts (symptoms) and iteratively derives new facts until a diagnosis is reached. This methodology is a direct application of the *Modus Ponens* inference rule, a cornerstone of propositional logic [2].

II. System Architecture

The architecture of our expert system is designed for clarity and modularity, separating the domain knowledge from the reasoning mechanism. This design choice, common in expert systems [1], allows for easier maintenance and expansion. The system consists of three primary components, as illustrated in Fig. 1.

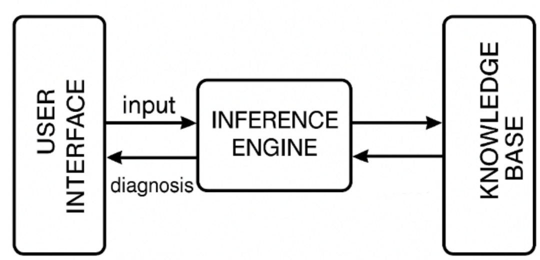

Fig. 1. High-level architecture of the Medical Diagnosis Expert System, showing the interaction between the User Interface, Inference Engine, and Knowledge Base.

- **Knowledge Base (KB):** This component serves as the repository for all domain-specific knowledge. It stores information in two forms:

- **Facts:** Assertions that are known to be true. In our system, facts represent the patient's symptoms (e.g., 'fever', 'cough').
- **Rules:** 'IF-THEN' constructs that define logical relationships between facts. These are modeled as Horn Clauses, which are a staple of logic programming [2].

- **Inference Engine:** This is the processing core of the system. It systematically applies the rules from the KB to the known facts to infer new information. Our engine implements a forward chaining strategy.
- **User Interface (UI):** A simple command-line interface that facilitates interaction. It allows the user to input initial facts (symptoms) and receives the final conclusions (diagnoses) from the system.

III. Implementation Details

The system is implemented in Python 3, with a focus on object-oriented design to encapsulate the logic of each architectural component.

A. Knowledge Base

The 'KnowledgeBase' class manages a set of facts and a list of rules. Facts are stored in a Python 'set' for efficient lookup and to prevent duplicates. Rules are stored as dictionary objects, each containing a list of 'conditions' and a single 'conclusion'.

A typical rule in our system follows the logical form of a definite clause [2]:

$$(S_1 \wedge S_2 \wedge \cdots \wedge S_n) \to D \tag{1}$$

where $S_1, \ldots, S_n$ are symptoms (conditions) and D is the resulting disease (conclusion). For example, the rule for diagnosing influenza is represented as:

```
IF fever AND headache AND muscle pain AND fatigue
THEN influenza
```

B. Inference Engine

The 'InferenceEngine' class implements the forward chaining algorithm. The core 'infer()' method repeatedly iterates through the rules in the KB. For each rule, it checks if all of its conditions are present in the current set of facts. This process can be formally described as:

Given a set of facts F and a rule R where $R = (C \to H)$, if all conditions in the set $C = \{c_1, c_2, \ldots, c_n\}$ are a subset of the facts ($C \subseteq F$), then the conclusion H is added to the set of facts.

$$\frac{F, (c_1 \wedge \cdots \wedge c_n \to H)}{F \cup \{H\}} \quad \text{if } \{c_1, \ldots, c_n\} \subseteq F \tag{2}$$

This loop continues until a full pass over the rule set produces no new facts, at which point the system has reached a stable state and the inference process terminates.

IV. Preliminary Results & Discussion

The prototype is fully functional and capable of diagnosing a set of predefined diseases based on user-provided symptoms. For instance, if a user inputs the facts 'fever', 'cough', and 'sore throat', the inference engine will fire the corresponding rule and add 'common cold' to the fact base, presenting it as the final diagnosis.

The system's primary strength lies in its transparency. The rule-based nature means its diagnostic path is fully traceable, unlike "black box" machine learning models. However, the current implementation has several limitations when compared to more advanced systems like the one proposed by Huang et al. [1]:

1) **Lack of Uncertainty:** Our system is deterministic and does not handle uncertainty. The research in [1] incorporates Certainty Factors (CF) to manage ambiguous information, a crucial feature for real-world medical applications.
2) **Simple Inference:** We use forward chaining, which is effective but can be inefficient if the goal is known. The system in [1] uses backward chaining, a goal-driven approach that is often more suitable for diagnostic tasks.
3) **Basic UI:** Our command-line interface is functional for testing but lacks the user-friendliness of a GUI, which would include features like explanation facilities and knowledge base editors [1].

V. Conclusion & Future Work

We have successfully designed and implemented a foundational expert system for medical diagnosis. The system correctly applies logical inference rules to a knowledge base to derive diagnoses from symptoms, fulfilling the core objectives of the project. The modular architecture provides a solid platform for future development.

Future work will focus on addressing the current limitations and incorporating more advanced features inspired by existing research [1]. Key priorities include:

- Implementing a mechanism for handling uncertainty, such as Certainty Factors or a Bayesian network approach.
- Exploring the implementation of a backward chaining inference engine.
- Developing a graphical user interface (GUI) to improve usability and to visualize the reasoning process.

These enhancements will move the project from a functional prototype to a more robust and practical diagnostic tool.

References

[1] X. Huang, X. Tang, and W. Zhang, "Research on Knowledge based Expert System for Medical Diagnosis," arXiv:2110.04439v1 [cs.AI], Oct. 2021.

[2] "CSE 440: Artificial Intelligence, Course Lecture Notes," Department of Computer Science and Engineering, University Name, 2024.

[3] Google, "Gemini Large Language Model," AI Development Tool, 2024. Accessed: Jul. 2, 2025.