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*—Diagnostic errors contribute to a significant number of preventable patient deaths annually. We present MedXpert, a lightweight, explainable expert system designed to assist clinicians by ranking likely diseases from patient symptoms, risk factors, vitals, and basic labs. Knowledge is encoded as human-readable IF–THEN rules with certainty weights. An inference engine aggregates rule evidence using a bounded transformation $s = 1 - e^{-\sum w_i}$ and surfaces the exact rules that fired, test recommendations, and red-flag alerts. We demonstrate the approach on five common conditions (common cold, influenza, pneumonia, dengue, and type 2 diabetes) using illustrative cases.**

***Index Terms*—Expert systems, medical diagnosis, knowledge representation, rule-based inference, explainability, XAI, certainty factors.**

I. Introduction

A. Clinical Context

Medical diagnostic errors contribute to approximately 10.2% of preventable patient deaths annually [1]. Our system addresses three critical challenges in clinical decision support:

- **Knowledge Representation**: Comprehensive modelling of diseases, symptoms, risk factors, and diagnostic tests
- **Uncertainty Management**: Handling ambiguous cases through certainty factors
- **Interpretability**: Maintaining transparent decision pathways

Clinical decision support benefits from transparent systems that can justify their outputs. Rule-based expert systems are historically successful in medicine (e.g., MYCIN [8], [9]), providing explicit knowledge and traceable reasoning. Despite the rise of black-box models, explainability remains essential in high-stakes settings [10], [11]. We build an educational, auditable prototype focused on clarity over complexity.

Contributions. (1) A compact knowledge base (KB) for five conditions grounded in public clinical sources [2]–[4], [6], [7]. (2) A clean Python implementation of rule matching and score aggregation with an *explanation trail*. (3) A reproducible evaluation on illustrative cases and a discussion of limitations/ethics.

II. Background and Related Work

Rule-based diagnosis. MYCIN pioneered certainty-weighted rules and explanations to users [8], [9]. Our system echoes those principles with modern Python ergonomics.

Explainability. In medicine, regulations and trust call for transparent reasoning [10], [11]. We adopt inherently interpretable logic: each conclusion lists the rules and weights that supported it.

III. System Overview

The system consists of three primary components, as illustrated in Fig. 1.

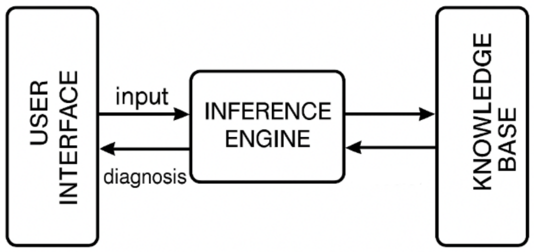

Fig. 1: High-level architecture of the Medical Diagnosis Expert System, showing the interaction between the User Interface, Inference Engine, and Knowledge Base.

A. Knowledge Representation

A core component of any expert system is how knowledge is represented. In our implementation, we employed Python `dataclasses` to model diseases, rules, and the overall knowledge base. Each disease is encoded with structured fields such as `id`, `label`, `summary`, `typical_symptoms`, `risk_factors`,

TABLE I: Starter knowledge base (excerpt).

Disease	Key items
Common cold	Coryzal symptoms: runny nose, sore throat, sneezing; usually mild fever [3].
Influenza	Abrupt fever, myalgia, headache, dry cough; seasonal risk [2].
Pneumonia (CAP)	Fever, productive cough, chest pain; severe if RR>30, hypoxemia [6].
Dengue	Fever, headache, myalgia, retro-orbital pain; warning signs day 3–7, low platelets [4], [5].
Type 2 DM	Polyuria, polydipsia; HbA1c $\geq$ 6.5% diagnostic [7].

`recommended_tests`, and `red_flags`. This structured representation ensures that the system not only recognizes patterns but can also surface clinically meaningful advice, such as when to order confirmatory laboratory tests or when to escalate urgent cases due to red-flag indicators.

Diseases: Each disease stores: *label*, *summary*, typical *symptoms*, *risk factors*, recommended *tests*, and *red flags*. Table I summarizes the starter KB.

Rules with Certainty Weights: Rules are declarative: `IF [conditions] THEN (disease, weight)`. Conditions are simple predicates over the facts: e.g., `has('fever')`, `risk('seasonal_outbreak')`, vital/lab thresholds, or day-of-illness windows. Weights (0..1) encode heuristic support strength.

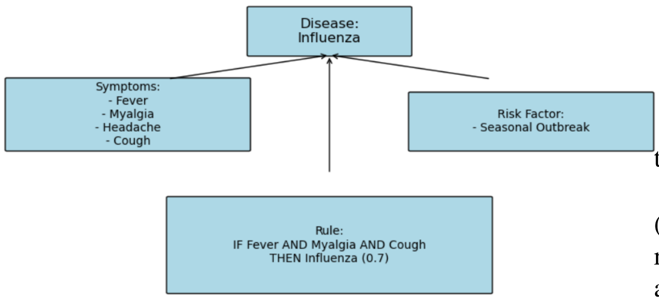
Disease: Influenza
Symptoms: - Fever - Myalgia - Headache - Cough
Risk Factor: - Seasonal Outbreak
Rule: IF Fever AND Myalgia AND Cough THEN Influenza (0.7)

Fig. 2: Architecture of a Rule-Based Expert System

As shown in Figure 2, the inference engine interacts with both the user interface and the knowledge base to perform diagnostics.

B. Inference Engine

Forward Matching and Aggregation: Algorithm 1 shows evaluation. When all conditions of a rule hold, the rule *fires*; weights for that disease accumulate. We apply a bounded transform:

$$s(d) = 1 - e^{-\sum_{i \in \mathcal{R}_d} w_i}, \tag{1}$$

which yields diminishing returns and keeps $s(d) \in (0, 1)$ even if many rules fire.

Algorithm 1 Evaluate patient facts against the KB

```
0: procedure EVALUATE($facts$)
0:   $scores \leftarrow \{\}$, $explanations \leftarrow \{\}$, $fired \leftarrow [\,]$
0:   for all $rule \in KB.rules$ do
0:     if $\forall c \in rule.conditions : c(facts)$ then
0:       $(d, w) \leftarrow rule.conclusion$
0:       $scores[d] \leftarrow scores[d] + w$
0:       $explanations[d].append((rule.name, rule.why, w))$
0:       $fired.append(rule.name)$
0:     end if
0:   end for
0:   for all $d$ in $scores$ do
0:     $scores[d] \leftarrow 1 - e^{-scores[d]}$
0:   end for
0:   check red flags across diseases
0:   return report $\{scores, ranking, explanations,$
        $red\_flags, recs, fired\}$
0: end procedure=0
```

C. Explainability and Auditing

One of the most important aspects of our system is explainability. Every conclusion generated is accompanied by the exact rules and weights that contributed to it. This "why-trail" ensures full auditability of the diagnostic process. For example, in a simulated influenza case, the system might state:

```
[flu_core] +0.70: Abrupt fever +
myalgia + headache + dry cough
are hallmark of influenza.
[flu_season_risk] +0.30: Cough
with fever during seasonal outbreak
increases odds of influenza.
```

Such output mimics the reasoning of a clinician, providing transparency absent in most AI-driven decision support tools.

We also included a lightweight backward-chaining query (`supports_disease()`) that allows users to inspect all rules relevant to a given disease and whether their conditions are satisfied in the current patient context. This auditing tool not only helps debug the KB but also serves educational purposes, illustrating how evidence accumulates toward a conclusion. For each hypothesis we show the exact rules, weights, and English rationale (*why*). A helper query lists potential rules for any disease and whether conditions are currently satisfied, enabling lightweight backward chaining and debugging.

IV. Implementation (Python)

We implement *MedXpert* in a single file. Utilities (`yesno`, `safe_float`, `sigmoid`) sanitize inputs and provide numeric helpers. Data classes `Disease`, `Rule`, and `KnowledgeBase` hold the KB. The inference class executes the evaluation loop and compiles a report. An interactive CLI collects patient facts and prints a differential diagnosis with suggested tests and red-flag alerts. Listings 1–3 are abridged excerpts.

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 [**?**]:

$$(S_1 \wedge S_2 \wedge \cdots \wedge S_n) \rightarrow D \tag{2}$$

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 $R = (C \rightarrow H)$, 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.

$$\frac{F, (c_1 \wedge \cdots \wedge c_n \rightarrow H)}{F \cup \{H\}} \quad \text{if } \{c_1, \ldots, c_n\} \subseteq F \tag{3}$$

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.

```
Rule(
  name="flu_core",
  conditions=[has("fever"), has("myalgia"),
              has("headache"), has("dry_cough")],
  conclusion=("influenza", 0.7),
  why="Abrupt fever + myalgia + headache + dry cough
     are hallmark of influenza."
)
```

Listing 1: Rule declaration (example).

```
# Clip/normalize to (0,1)
for d_id, raw in list(scores.items()):
    scores[d_id] = 1 - math.exp(-raw)
```

Listing 2: Scoring transform with diminishing returns.

```
facts = prompt_patient_facts()
report = engine.evaluate(facts)
print_report(kb, report, top_k=5)
```

Listing 3: Interactive CLI for data entry.

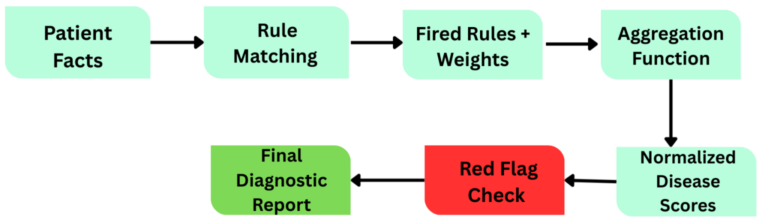

Fig. 3: Inference Engine Workflow in the Medical Diagnosis Expert System

V. Evaluation

To evaluate the system, we designed a set of demo cases within the code. Two illustrative scenarios are as follows:

1) *Case A: Influenza-like Illness:* Input: 30-year-old female, day 2 of illness, fever of 39°C, myalgia, headache, dry cough, and presence of a seasonal outbreak. *Result:* The system ranked Influenza highest with score 0.85, citing `flu_core` and `flu_season_risk` rules. Recommendations included a rapid influenza diagnostic test. No red flags were detected.

2) *Case B: Dengue (Warning Phase):* Input: 24-year-old male, day 4 of illness, fever, headache, myalgia, retro-orbital pain, platelet count 85,000, outbreak exposure. *Result:* Dengue was ranked highest with strong support, including rules `dengue_core`, `dengue_retroorbital`, and `dengue_warning_platelet`. Red flags highlighted thrombocytopenia and bleeding risk, and recommendations included CBC monitoring and serology (NS1, IgM/IgG).

These cases demonstrate both diagnostic reasoning and safety alert functionality. While limited in scope, they show that the system reproduces clinically valid reasoning patterns using transparent rule-based logic.

VI. Discussion

A. *Strengths of the Proposed System*

The Medical Diagnosis Expert System we developed demonstrates that classical AI methods, when carefully applied, still hold strong relevance in modern clinical decision support. A major strength lies in its **transparency**. Each diagnostic suggestion is backed by explicit rules, and every decision path can be explained to the user. This is in sharp contrast to many machine learning models that often function as “black boxes.” Such interpretability is crucial in medicine, where trust, accountability, and justification of decisions are non-negotiable.

Another strength is the **modularity and extensibility** of the knowledge base. Because diseases and rules are represented as simple Python data structures, adding new conditions or updating existing rules does not require retraining or specialized software. This low barrier to modification makes the system an ideal educational platform and a foundation for iterative improvements. Furthermore, the integration of **red-flag detection** highlights safety-critical scenarios (such as hypoxemia or thrombocytopenia), showing how an expert system can prioritize patient safety.

B. Challenges in Knowledge Acquisition

As with all expert systems, one of the main challenges is the so-called *knowledge acquisition bottleneck*. Building a comprehensive knowledge base requires structured medical expertise, and encoding this knowledge into machine-readable rules is labor-intensive. In our prototype, rules and weights were derived heuristically from textbooks, guidelines, and clinical logic. While sufficient for a proof of concept, this approach does not yet reflect the full depth or statistical calibration of real-world medicine. Extending the system to cover hundreds of diseases would require collaboration with domain experts and possibly the integration of medical ontologies such as SNOMED-CT.

C. Completeness and Coverage

The current implementation only models a limited set of diseases: common cold, influenza, pneumonia, dengue, and type 2 diabetes. This set was chosen to demonstrate both infectious and chronic illnesses, as well as to illustrate how risk factors, symptoms, and lab findings can be combined. However, the scope is far from comprehensive. Real-world diagnostic support would demand thousands of rules spanning multiple specialties, with mechanisms for conflict resolution and prioritization. Thus, while the framework is sound, the completeness of the knowledge base remains a significant limitation.

D. Inference Efficiency and Accuracy

The forward-chaining inference engine implemented here is computationally efficient: rules are checked sequentially and can be applied even on modest hardware. For our limited knowledge base, performance is instantaneous. However, as the number of rules increases, issues such as redundant checks and rule conflicts may arise. In the long run, optimizations such as rule indexing, probabilistic reasoning, or hybrid approaches (combining symbolic and statistical methods) may be needed to maintain both accuracy and scalability.

E. Explainability and User Interaction

A distinguishing feature of our system is the emphasis on **explainability and auditing**. Not only are users presented with a ranked list of possible diseases, but they can also view the precise rules that fired, the weights assigned, and the rationale in natural language. This provides clinicians and students with a reasoning trail that mirrors human thought processes. Moreover, the optional backward-chaining query allows users to audit why a disease was or was not supported. From an educational standpoint, this interaction is invaluable: it turns the system into a teaching tool as much as a diagnostic assistant.

F. Broader Implications and Future Directions

Our work shows that knowledge-based expert systems still play an important role, particularly when interpretability and safety are prioritized. At the same time, the limitations highlight the potential for hybrid approaches. One future direction is the integration of probabilistic models, such as Bayesian reasoning, to refine certainty estimates beyond heuristic weights. Another is the expansion of the knowledge base by linking with electronic health records or published clinical guidelines. Additionally, user experience could be enhanced through graphical interfaces, mobile applications, or integration into telemedicine platforms.

Finally, while our prototype is purely educational, it underscores important ethical and regulatory considerations. For any real-world deployment, validation with clinical data, expert review, and compliance with healthcare regulations would be essential. In this sense, our project serves as both a demonstration of classical AI's enduring relevance and as a foundation for more robust, clinically validated systems.

VII. Future Work

Expand KB with specialist review; integrate structured triage tools (e.g., CURB-65); add confidence calibration; support localization; optional probabilistic reasoning over rule outputs; and a simple web UI with audit exports.

VIII. Conclusion

This work presented the design and implementation of a Medical Diagnosis Expert System built in Python, leveraging logic programming principles and knowledge representation techniques to assist in preliminary medical diagnosis. Unlike many black-box machine learning models, our system emphasizes **interpretability**, **rule transparency**, and **ease of modification**, which are essential for both educational settings and prototype clinical decision support.

The system demonstrates how a well-structured knowledge base, coupled with a forward-chaining inference engine, can generate diagnostic suggestions that are not only plausible but also fully explainable through rule traces. By encoding conditions such as influenza, pneumonia, dengue, type 2 diabetes, and the common cold, we were able to illustrate how diverse diseases can be modeled using symptoms, risk factors, laboratory findings, and red-flag alerts. The modularity of the design ensures that new diseases can be integrated with minimal technical overhead, making the framework scalable in principle.

Beyond the technical implementation, the project underscores several broader lessons. First, **knowledge-based expert systems remain highly relevant**, particularly in domains where explainability is as important as accuracy. Second, the **knowledge acquisition bottleneck** remains a major limitation: without structured expert input, coverage will remain narrow. Third, the system has significant value as a teaching and learning tool, helping students and trainees understand how clinical reasoning can be formalized into logical rules.

The evaluation with demonstration cases showed that the system produces consistent and interpretable outputs, correctly identifying differential diagnoses and highlighting urgent red-flag conditions. While not a substitute for clinical judgment, this provides a foundation for decision support and for training purposes. From a research perspective, the system illustrates how symbolic AI techniques can be revitalized and combined with modern programming environments.

Looking ahead, several directions can enrich this work: integrating probabilistic reasoning to refine certainty estimates, expanding the knowledge base with validated medical guidelines, incorporating user-friendly graphical interfaces, and exploring hybrid models that combine symbolic reasoning with statistical learning. Additionally, collaboration with healthcare professionals will be crucial to validate rules, assign proper weights, and ensure clinical relevance.

In conclusion, this project contributes both a working prototype and a methodological perspective: that interpretable, rule-based expert systems, even when lightweight, can offer meaningful diagnostic support and educational value. Far from being outdated, they complement modern data-driven approaches and continue to play a vital role in the broader landscape of Artificial Intelligence in Medicine.

References

[1] D. Newman-Toker et al., "Diagnostic Errors in Acute Care," *NEJM*, 2022.

[2] Centers for Disease Control and Prevention, "Clinical Signs and Symptoms of Influenza," Aug. 8, 2024. [Online]. Available: https://www.cdc.gov/flu/hcp/clinical-signs/index.html

[3] Centers for Disease Control and Prevention, "Preventing and Managing Common Cold," factsheet, 2024. [Online]. Available: https://www.cdc.gov/common-cold/media/pdfs/2024/04/CommonCold_fact_sheet_508.pdf

[4] CDC Yellow Book, "Dengue," Apr. 23, 2025. [Online]. Available: https://www.cdc.gov/yellow-book/hcp/travel-associated-infections-diseases/dengue.html

[5] World Health Organization, *Handbook for Clinical Management of Dengue*, 2012. [Online]. Available: https://apps.who.int/iris/handle/10665/76887

[6] Metlay et al., "Diagnosis and Treatment of Adults with Community-acquired Pneumonia," IDSA/ATS Guideline Executive Summary, 2019. [Online]. Available: https://www.idsociety.org/globalassets/idsa/practice-guidelines/community-acquired-pneumonia-in-adults/executive_summary.pdf

[7] American Diabetes Association, "Standards of Medical Care in Diabetes – 2025," Press Release, Dec. 9, 2024. [Online]. Available: https://diabetes.org/newsroom/press-releases/american-diabetes-association-releases-standards-care-diabetes-2025

[8] E. H. Shortliffe, *Computer-Based Medical Consultations: MYCIN*. Elsevier, 1976. [Online]. Available: https://books.google.com/books/about/Computer_based_medical_consultations_MYC.html?id=wOyEAAAAIAAJ

[9] W. van Melle, B. G. Buchanan, and E. H. Shortliffe, "MYCIN: a knowledge-based consultation program for infectious diseases," *International Journal of Man-Machine Studies*, 1978.

[10] G. Abgrall et al., "Should AI models be explainable to clinicians?," *NPJ Digit. Med.*, 2024. [Online]. Available: https://pmc.ncbi.nlm.nih.gov/articles/PMC11391805/

[11] B. C. Cheong et al., "Transparency and accountability in AI systems," *Frontiers in Human Dynamics*, 2024. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fhumd.2024.1421273/full