AI-Powered Rule-Based Expert System in Prolog for Medical Diagnosis

Overview

This is a fully functional expert system in Prolog designed to diagnose diseases based on user symptoms. It showcases:

- Rule-based reasoning
- Pattern matching
- Backtracking
- Decision trees
- Dynamic knowledge base handling
- Natural Language Processing (NLP)-style prompts

This type of system is used in AI-driven chatbots, clinical decision-making, and even robotic cognition systems.

Features

- Symptom-based questioning
- Dynamic knowledge acquisition
- Rule chaining
- Easy scalability for new rules
- Semantic pattern matching using Prolog logic

Full Code with Comments

```
% -------------------------------
% Prolog Expert System: Medical Diagnosis
% -------------------------------
% Author: Maria | Technical Writer & AI Coder
% Description: Rule-based system that suggests a possible disease based on patient
symptoms.
% --------------------------------------------------------

:- dynamic(symptom/1).
:- dynamic(asked/1).
```

```
start :-
    write('□ Welcome to the AI Medical Diagnosis Expert System!'), nl,
    write('Please answer the following questions with yes. or no.'), nl, nl,
    retractall(symptom(_)),
    retractall(asked(_)),
    diagnose(Disease),
    nl, write('□ Based on the symptoms, the system suggests you might have: '),
write(Disease), nl,
    nl, write('□ This is just an AI-based suggestion. Please consult a human doctor for
final confirmation.'), nl.

start :-
    nl, write('□  Sorry, the system could not determine a diagnosis based on your
responses.'), nl,
    write('Try again or consult a medical professional.'), nl.

diagnose(flu) :-
    verify(fever),
    verify(headache),
    verify(body_ache),
    verify(sore_throat),
    verify(runny_nose),
    verify(cough).

diagnose(common_cold) :-
    verify(runny_nose),
    verify(sore_throat),
    verify(sneezing),
    verify(cough).

diagnose(malaria) :-
    verify(fever),
    verify(chills),
    verify(sweating),
    verify(headache),
    verify(nausea).

diagnose(covid19) :-
    verify(fever),
    verify(cough),
    verify(shortness_of_breath),
    verify(loss_of_taste_or_smell),
    verify(fatigue).

diagnose(migraine) :-
    verify(headache),
    verify(nausea),
    verify(sensitivity_to_light),
    verify(blurred_vision).

verify(Symptom) :-
    symptom(Symptom).

verify(Symptom) :-
    \+ asked(Symptom),
    ask(Symptom).

ask(Symptom) :-
    write('Do you experience the symptom: '), write(Symptom), write('? (yes./no.)'), nl,
    read(Response),
    asserta(asked(Symptom)),
    (Response == yes -> asserta(symptom(Symptom)); true),
    Response == yes.

reset :-
    retractall(symptom(_)),
    retractall(asked(_)),
    write('□ System has been reset.'), nl.

/
Sample Usage:
?- [medical_diagnosis].
?- start.
?- reset.
?- halt.
*/

% Extensions:
% diagnose(diabetes) :-
%     verify(frequent_urination),
%     verify(thirst),
%     verify(weight_loss),
%     verify(blurred_vision).

nlp_question(Symptom, Sentence) :-
    symptom_mapping(Symptom, Sentence).

symptom_mapping(fever, 'Do you have a high body temperature or fever?').
symptom_mapping(cough, 'Are you experiencing frequent coughing?').
symptom_mapping(sore_throat, 'Do you have a sore or scratchy throat?').
symptom_mapping(shortness_of_breath, 'Are you feeling shortness of breath or difficulty
breathing?').
symptom_mapping(loss_of_taste_or_smell, 'Have you lost your sense of taste or smell?').

ask_nlp(Symptom) :-
    nlp_question(Symptom, Sentence),
    write(Sentence), write(' (yes./no.)'), nl,
    read(Response),
    asserta(asked(Symptom)),
    (Response == yes -> asserta(symptom(Symptom)); true),
    Response == yes.
```

Real-World Applications

- Used in AI chatbots in healthcare
- Embedded in robotic assistants
- Foundation of rule engines in enterprise systems
- Can be linked with voice UIs for voice-based symptom checking
- Base framework for IoT sensor rule validation

GitHub Optimization (README.md sample)

AI Medical Diagnosis Expert System in Prolog

High-performance rule-based expert system using Prolog, simulating a medical diagnostic AI agent. Built with pattern matching, dynamic memory, and natural language prompts.

Key Features
- Interactive Q&A interface
- Diagnoses based on logical inference
- Dynamic symptom tracking
- Modular disease definition
- Easily extensible

Run Instructions
1. Install SWI-Prolog
2. Load the file:

```
?- [medical_diagnosis].
?- start.
```

Use Cases
- Chatbot reasoning backend
- Healthcare simulation apps
- NLP + logic hybrid testing
- Interview technical demonstration