**Integrative Task II**

**DrAutoBot - Expert System for Electric Car Diagnostics**

**Abstract**

This report outlines the development and implementation of DrAutoBot, an expert system-based chatbot designed to diagnose issues in electric vehicles. Built using Python's experta library, DrAutoBot combines rule-based reasoning with the simplicity of a conversational interface deployed on Telegram. The system identifies problems in six primary areas of electric vehicle functionality—battery, motor, braking, electronics, tires & suspension, and software—and provides diagnostics based on user-provided answers to predefined yes/no questions.

**Introduction**

**Context**

The transition to electric vehicles has introduced unique challenges for vehicle maintenance and diagnostics. Traditional mechanics may lack expertise in their specific issues, creating a demand for automated systems that can assist owners in identifying problems efficiently. Expert systems, known for their ability to emulate human reasoning, are ideal for this purpose. DrAutoBot uses this technology to assist users in diagnosing their vehicle problems via a conversational chatbot interface.

**Objectives**

The primary objectives of the project are:

1. To create an intelligent diagnostic tool for electric vehicles using expert systems.
2. To ensure ease of access by deploying the chatbot on the Telegram application.
3. To provide accurate diagnostics across six core areas of vehicle functionality.

**Methodology**

**Bayesian Network**

The Bayesian Network was constructed to model potential issues in an electric vehicle's systems and their observable symptoms. The methodology involved defining key subsystems such as the battery, motor, braking, electronics, tires and suspension, software, and miscellaneous components as nodes in the network. Symptoms related to each subsystem were added as dependent nodes, creating edges to represent causal relationships.

Conditional Probability Distributions (CPDs) were defined for each subsystem and its symptoms, assigning probabilities to model the likelihood of issues causing specific symptoms. These probabilities were either based on domain knowledge or assumptions for simulation purposes.

The structure ensures the network captures dependencies between vehicle systems and symptoms, enabling probabilistic inference using algorithms like Variable Elimination. This approach supports diagnostics by computing the likelihood of underlying issues given observed symptoms.

**Expert System**

The expert system is implemented using the experta library, which facilitates the creation of knowledge-based systems. The system uses:

* **Knowledge Base:** Predefined rules and facts related to EV diagnostics.
* **Inference Engine:** Logical reasoning to derive diagnostics based on user inputs.
* **Areas of Focus:** The system addresses six diagnostic areas:
  1. **Battery Issues**: Charging, discharging, and cell health.
  2. **Motor Issues**: Performance, overheating, and connectivity.
  3. **Braking Issues**: Regenerative braking and mechanical failure.
  4. **Electronics Issues**: Malfunctions in lights, screens, and sensors.
  5. **Tires & Suspension**: Wear, alignment, and pressure.
  6. **Software Issues**: System errors, firmware, and connectivity problems.

**Rule-Based Logic**

The knowledge base contains rules to evaluate combinations of user responses. In this project, default questions and facts are stored within its structure to be later implemented (depending on user input) by the rules that were previously designed with the support of the Experta library.

**User Interaction**

After a brief setup of the user’s device of choice to select the chatbot, the user interacts with DrAutoBot via a series of yes/no questions. Based on this input which traduces into boolean responses, the system evaluates relevant rules and generates a diagnostic on the issues that were asked about.

**Telegram Integration**

The chatbot interface is deployed on Telegram using the telebot library. Key features include:

* **User-Friendly Interaction:** Users can type yes or no responses in a chat-like interface.
* **Real-Time Diagnostics:** Responses are processed instantly, and diagnostics are returned within seconds.
* **Multi-Area Diagnosis:** Users can specify multiple problem areas for comprehensive analysis.

**Implementation**

**Tools and Technologies**

1. **Python**: Primary programming language.
2. **Bayesian Network & Tabular CPD**: For building the Bayesian network.
3. **Experta**: For building the expert system.
4. **Telebot**: For Telegram chatbot integration.
5. **Telegram Bot API**: To deploy and manage the chatbot.

**Product Flow**

1. **Initialization**: The bot prompts the user to select areas of concern (e.g., battery, motor).
2. **Data Collection**: Users answer yes/no questions about symptoms.
3. **Inference**: The expert system applies rules based on user responses.
4. **Output**: Diagnostics are displayed in the Telegram chat.

**Conclusion**

DrAutoBot demonstrates the effectiveness of combining expert systems with chatbot technology for specialized applications like EV diagnostics. Its rule-based logic ensures reliable outputs, while the Telegram interface makes the tool accessible to a broad audience. Future enhancements may include:

* Integration of natural language processing for more conversational interactions.
* Expansion of the knowledge base to cover additional areas and scenarios.