scientific reports

Check for updates

OPEN

Harnessing AI forward and backward chaining with telemetry data for enhanced diagnostics and prognostics of smart devices

Muhammad Shoaib Farooq[1], Rizwan Pervez Mir[1], Atif Alvi[1], Kilian Tutusaus[2,3,4], Eduardo Garcia Villena[2,5,6], Fadwa Alrowais[7]✉, Hanen Karamti[7] & Imran Ashraf[8]✉

In the rapidly evolving landscape of artificial intelligence (AI) and the Internet of Things (IoT), the significance of device diagnostics and prognostics is paramount for guaranteeing the dependable operation and upkeep of intricate systems. The capacity to precisely diagnose and preemptively predict potential failures holds the potential to considerably amplify maintenance efficiency, diminish downtime, and optimize resource allocation. The wealth of information offered by telemetry data gathered from IoT devices presents an opportunity for diagnostics and prognostics applications. However, extracting valuable insights and making well-timed decisions from this extensive data reservoir remains a formidable challenge. This study proposes a novel AI-driven framework that integrates forward chaining and backward chaining algorithms to analyze telemetry data from IoT devices. The proposed methodology utilizes rule-based inference to detect real-time anomalies and predict potential future failures, providing a dual-layered approach for diagnostics and prognostics. The results show that the diagnostics engine using forward chaining detects real-time issues like "High Temperature" and "Low Pressure," while the prognostics engine with backward chaining predicts potential future occurrences of these issues, enabling proactive prevention measures. The experimental results demonstrate that adopting this approach could offer valuable assistance to authorities and stakeholders. Accurate early diagnosis and prediction of potential failures have the capability to greatly improve maintenance efficiency, minimize downtime, and optimize cost.

Keywords Internet of Things, Telemetry data, Diagnostics and prognostics, Forward chaining, Reverse chaining

In today's quickly developing digital landscape, wide availability of Internet of Things (IoT) devices has accompanied a new era of interconnectedness that has fundamentally transformed our daily lives[1]. These gadgets, which range from industrial gear to smart appliances, have ushered in a new age of efficiency and ease. However, as the IoT grows, a critical problem arises: how can we make these gadgets smarter and more independent, especially in terms of prognostics and diagnostics? This study embarks on a journey to explore this innovative approach to address the evolving needs of device management and maintenance.

The coming together of IoT and artificial intelligence (AI) offers a fantastic chance to give intelligence to IoT devices. The International Data Corporation (IDC) estimates the number of IoT devices to be 41.6 billion by 2025 underscoring the urgent need for creative methods of managing and maintaining these devices. By integrating forward chaining and backward chaining, two different reasoning techniques, this work seeks to maximize the potential of artificial intelligence[2]. While backward chaining begins with an outcome and works backward to discover possible causes, forward chaining makes predictions based on known data[3]. Even though

[1]Department of Computer Science, School of System and Technology, University of Management and Technology, Lahore 54000, Pakistan. [2]Universidad Europea del Atlantico, Isabel Torres 21, Santander 39011, Spain. [3]Universidade Internacional do Cuanza, Cuito, Angola. [4]Universidad de La Romana, La Romana, Dominican Republic. [5]Universidad Internacional Iberoamericana, Campeche 24560, Mexico. [6]Universidad Internacional Iberoamericana Arecibo, Puerto Rico 00613, USA. [7]Department of Computer Sciences, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, P.O. Box 84428, Riyadh 11671, Saudi Arabia. [8]Department of Information and Communication Engineering, Yeungnam University, Gyeongsan 38541, Republic of Korea. ✉email: FMAlRowais@pnu.edu.sa; imranashraf@ynu.ac.kr

these methods have been extensively used in expert and knowledge-based systems, there is still much to learn about how to use them in conjunction with telemetry data for device prognostics and diagnostics. Figure 1shows the devices and sensors used in telemetry data[4].

The utilization of device telemetry data and integrated information mining to create an enhanced health monitoring system is becoming a reality thanks to the data mining and machine learning paradigms[5]. Device telemetry data is paramount information and can be utilized to detect anomalies in the device in a timely manner and preventive measures can be taken in advance. More than ever, there is a need for enhanced device prognostics and diagnostics. Predicting and averting device malfunctions is not only economical but also vital in a number of fields, including industrial automation, infrastructure management, and healthcare. The goal of this research is to show how telemetry data and AI reasoning approaches may be used to unleash device diagnostic and prognostic potential, minimize downtime, and improve operational efficiency.

The main objective of this research is to propose and demonstrate how the combination of device telemetry data with AI reasoning techniques (forward and backward chaining) may provide novel approaches to device diagnostics and prognostics, therefore guaranteeing the long-term dependability and efficiency of devices. The novelty of this work lies in the development of applications to prove the hypothesis, as discussed in Section 3, during the design of this methodology and the analysis of existing solutions. Reasoning techniques, forward and backward chaining, have been used in a number of fields, including expert systems; however, there is still much to learn about how to use them in conjunction with telemetry data to diagnose and prognosticate IoT devices.

This research paper is structured as follows: The approaches utilized by other researchers are examined in Section 2, outlining the useful applications of forward chaining and backward chaining in telemetry data analysis. In Section 3, the proposed methodology has been analyzed comprehensively utilizing well-defined device telemetry data and inference rules. Then, the outcomes of the experiments with actual IoT devices telemetry data will subsequently be presented, along with the experimental setup in Section 4. The study is finalized by outlining how this discovery might revolutionize the field of device diagnostics and prognostics and provide some recommendations for future research directions in Section 5.

Literature reivew

Recent research has begun to employ artificial intelligence to diagnose devices and predict future anomalies. Device telemetry data is rich and unique and applying a methodology that has the ability to transform its processing against the input and generate alerts can be very useful in monitoring device health and in taking prompt actions. Let's discuss in detail a few of the endeavors of recent years in device diagnostics and prognostics with their strengths and weaknesses.

The study of Serkan Ayvaz et al. demonstrates how it may facilitate real-time analysis by utilizing machine learning techniques to quickly evaluate Internet of Things data and anticipate any maintenance concerns[6]. This immediateness makes it possible to respond to equipment problems quickly, thus decreasing downtime. Additionally, the use of IoT data guarantees a thorough comprehension of the health of the equipment, which may result in increased productivity and cost savings. However, the effectiveness of the system in making decisions in real time may be impacted by potential difficulties in managing the complexity of putting in place

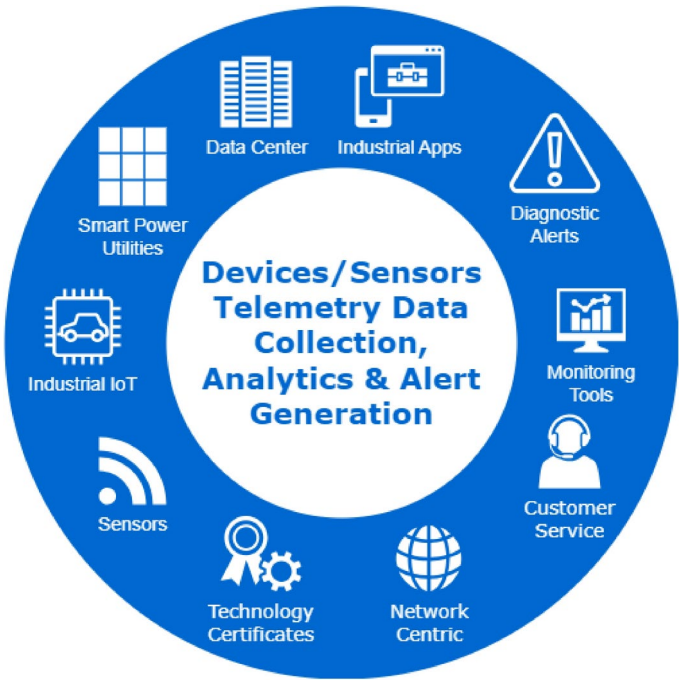


Fig. 1. Device telemetry data collection and diagnostic alert generation, adopted from[4].

and maintaining such a system. These difficulties include the need for abundant, high-quality data in order to make accurate predictions, the interpretability of machine learning models, and the system's reliance on consistent connectivity.

Zhou et al. demonstrate this specialization by offering a focused investigation of server system failure diagnosis through the use of forward chaining expert systems[7]. It provides a thorough understanding of the particular field of server diagnostics as well as useful approaches and techniques for locating and fixing errors. However, its narrow scope can be a drawback. It might miss information that could be obtained using backward chaining that could affect or influence server performance if artificial intelligence forward chaining is the exclusive focus. Furthermore, given the quick advancement of server technologies and failure scenarios, the diagnostic approach's flexibility or scalability may be restricted if it only uses third-party application programming interfaces (APIs).

A specific method designed for agricultural diagnostics is presented by Muhammad Saiful and Amri Muliawan Nur[8]. In order to provide farmers or agriculturalists with quick and convenient access to diagnostic data, a web-based expert system that uses forward chaining for accessibility and user-friendly diagnoses for corn plant diseases may be used. However, its reliance on the caliber and volume of incoming data, along with that backward chaining is not used that may help in identifying more facts based on available goals. The accuracy of diagnoses is also impacted by the extent of the knowledge base inside the system and the fluctuations in symptom presentation.

Cunsong Wang and colleagues demonstrate a strategic focus on the vital subject of in-orbit spacecraft health monitoring[9]. Utilizing telemetry data for diagnostic purposes exemplifies how data-driven analysis may be applied practically to anticipate device degradation, allowing for preventative maintenance and perhaps increasing the lifespan of the spacecraft. The focus on real-time monitoring may enable prompt actions by giving an instantaneous insight into the spacecraft's condition. On the other hand, potential drawbacks include issues with data amount and complexity, which might impair the accuracy of deterioration projections. The amount and diversity of telemetry data that is accessible may restrict the accuracy, which might have an impact on the system's capacity to predict irregular or uncommon deterioration patterns.

Anwar specializes in computer damage diagnostics using the forward chaining approach with an expert system. This method provides an organized and methodical approach to computer problem diagnosis, which may facilitate prompt and precise problem detection[10]. With its built-in knowledge base and rule-based reasoning, an expert system might be used to expedite the diagnostic process and provide consistent and dependable issue solving. However, the suggested fix limits the system's capacity to adapt to new or complicated computing problems. The breadth and depth of the knowledge base may limit the system's efficacy and make it more difficult for it to handle new or uncommon computer damage scenarios outside of its pre-existing rules or dataset. Furthermore, in the absence of backward chaining the limitation and effectiveness of the system could be contingent upon increasing problems to accommodate evolving computer technologies and damage patterns, which could pose a challenge in ensuring the system's continual relevance and reliability over time.

The study examines a number of current investigations in the field of device prognostics and diagnostics, highlighting the advantages and disadvantages of each strategy. Ayvaz et al.[6] show the possibility of real-time machine learning analysis using Internet of Things data, guaranteeing quick problem discovery but maybe posing difficulties in handling system complexity and data quality. Zhou et al.[7]emphasize the experience in the field of server diagnostics by focusing on it, however, their restricted emphasis restricts the insights that may be gained from other approaches, which could have an impact on server performance. The use of forward chaining for agricultural diagnostics is highlighted in the study by Saiful and Nur[8]. This method provides easy-to-understand diagnoses for illnesses of maize plants, but the lack of backward chaining and poor data quality may compromise accuracy. Wang and associates[9]concentrate on tracking the health of spacecraft while in orbit, emphasizing the value of telemetry data but also pointing out possible constraints on the quantity and variety of data that might compromise the accuracy of the system. Last but not least, Anwar is an expert in computer damage diagnostics, demonstrating the efficiency of forward chaining techniques in quick and accurate issue identification[10]. However, the lack of backward chaining and the difficulties in adjusting to changing computer technologies and damage patterns may limit the system's future relevance and flexibility.

Research gaps

The related work highlights significant advancements in device diagnostics and prognostics, but key research gaps remain. Most studies emphasize forward chaining methods without integrating backward chaining, which limits the ability to infer additional facts or anticipate potential anomalies effectively. Moreover, challenges such as managing data complexity, ensuring model interpretability, and adapting systems to evolving technologies or diverse domains persist. Limited attention has been given to integrating multi-chaining methods for enhanced flexibility and accuracy across different application domains.

Research objectives

- To check the device diagnostics using the forward chaining directly to identify patterns that indicate possible device failure.
- Check the device prognostics using backward chaining to prognosticate the future state or behavior of a device grounded on observed data and predefined rules.
- A quick review of telemetry data analysis for quick problem identification.

Contributions

This paper introduces a novel approach that combines forward and backward chaining to enhance real-time device diagnostics and prognostics. The proposed system addresses existing gaps by leveraging multi-chaining

to infer deeper insights from telemetry data, ensuring higher prediction accuracy and adaptability to evolving technological contexts.

Material and methods

This section presents the proposed framework for IoT device diagnostics and prognostics using telemetry data, powered by an AI inference engine with forward and backward chaining. This is basically an expert system developed in C# based that works on a flexible set of rules to parse telemetry data for various fault/warning diagnoses and prognostics. Forward chaining makes use of the available data and applies rules to deduce conclusions. On the other hand, backward chaining starts from the goal and drills down to a new fact. This engine can work on real time and offline data sources. Task Parallel Library (TPL) is used for the parsing of data in parallel. Selected data from the database is divided into multiple chunks based on the no of processors available and is processed in parallel for better performance and prompt intimation. The workflow of the proposed approach is shown in Figure 2.

Components

The following are the major components of this system:

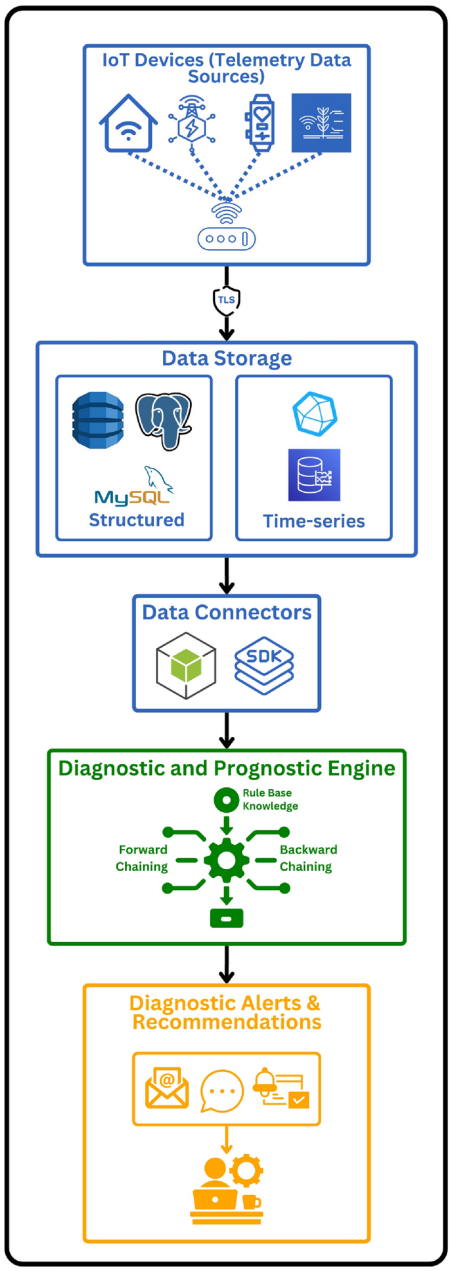


Fig. 2. Workflow diagram of the proposed approach.

- Knowledge Base or Expert System Engine Rules
- Telemetry Data: To be processed and examined by this system
- User Interface: Dashboard
- Actions: Intimation of the diagnostic alerts through email or SMS

Knowledge base
An expert system's knowledge base houses the information, rules, and logic necessary for problem-solving or decision-making[11]. It includes a database of data, heuristics, and facts that allow the system to simulate human reasoning. The guidelines or instructions that the system adheres to in order to draw inferences or make judgments are known as expert system engine rules, and they are an essential component of the knowledge base. By repeatedly applying these rules, the engine simulates human competence in a certain topic by evaluating an issue in light of the available facts and coming to conclusions or suggestions. The robustness and correctness of these rules are critical to the system's decision-making efficacy and accuracy.

Telemetry data
Information gathered remotely from diverse systems or IoT devices and usually sent wirelessly is referred to as telemetry data[12]. It offers instantaneous insights into these systems' behavior, performance, or condition. This information frequently consists of measurements, readings, or operational characteristics that support problem-solving, analysis, and monitoring. Telemetry data helps with diagnosis, predictive maintenance, and performance improvement in a variety of industries, including aircraft, healthcare, and industrial settings. The data that is gathered may include characteristics linked to health in medical devices as well as temperature and pressure readings in equipment. The evaluation and comprehension of this data are essential for making well-informed decisions, identifying irregularities, and implementing suitable measures to guarantee optimal performance and avert any problems.

User interface
A device diagnostics and prognostics system's user interface consists of an extensive dashboard with real-time alarms and data visualizations. Administrator users have access to dynamic graphs that help with issue detection and speedy decision-making by showcasing trends, anomalies, warnings, or certain performance indicators. Furthermore, the dashboard prominently displays live warnings and notifications to guarantee administrator users receive real-time updates on important concerns or deviations from usual operations. This configuration improves overall user experience and system usability by enabling effective monitoring, prompt reaction to abnormalities, and a clear, understandable presentation of diagnostic information.

Actions
The conclusions or suggestions that are generated from this system's reasoning process are called actions. These functions aim to notify the user of possible problems, offer advice, make recommendations, or automatically carry out a problem-solving solution. Following the processing and implementation of pre-established rules, results from the analysis of the telemetry data are used to produce actions. The actions can be generated in the form of Simple Message Service (SMS), Simple Notification Server (SNS), and Emails to the designated users with related information. These actions can be configured according to the requirements and scenarios and assign recipients accordingly. The effectiveness of these operations establishes the system's capacity to address problems, render precise judgments, and offer users insightful direction.

Process flow

Figure 3 depicts a process flow of how data is collected from various sources (real-time and offline) such as smart grid substations, smart healthcare units, smart house devices, and smart farming is stored in a variety of database sources. The diagnostic engine (Inference Engine) can use a variety of data connectors to connect different data types of data sources available e.g., MySQL Connector to connect MySQL database, similarly AWS DynamoDB SDK to connect AWS cloud database, and so on as shown in figure 2. Moreover, when the data source is connected with the inference engine, the knowledge base or rules can be written/updated in accordance with the data. Finally, the data is parsed under rules and diagnostic alerts can be generated which may help administrators and authorities in taking preemptive actions beforehand. Following Figure 4 is the layered architecture of the proposed technique for IoT device diagnostics and prognostics using telemetry data powered by an AI inference engine with forward and backward chaining. Let's discuss each layer of the architecture in depth.

In *layer 1* there is a depiction of IoT devices (sensors) that are installed and configured to their respective gateways and to their clouds/virtual private servers/dedicated servers. These IoT devices (sensors) are connected to their gateways through WIFI or BLE and send their heartbeats and sensor-related information in the form of telemetry data to their respective databases over TLS-based secure communication channels using Message Queuing Telemetry Transport (MQTT) and other protocols.

In *layer 2*, the telemetry data is stored in different types of databases according to their requirements and design. Devices information is usually stored in a structured database such as MySQL, SQL Server, DynamoDB, and PostgreSQL according to the specific requirements. Whereas, the telemetry data is stored in an unstructured way in a time-series database such as InfluxDB and Timestream by Amazon Web Services (AWS).

In *layer 3*, different types of data connectors are shown in the figure above. These connectors are used by the diagnostic inference engine to connect to the desired databases to collect telemetry data using SDKs and required connection details provided by the respective providers e.g., DynamoDB and AWS IoT SDKs are available for Android, iOS, and C#

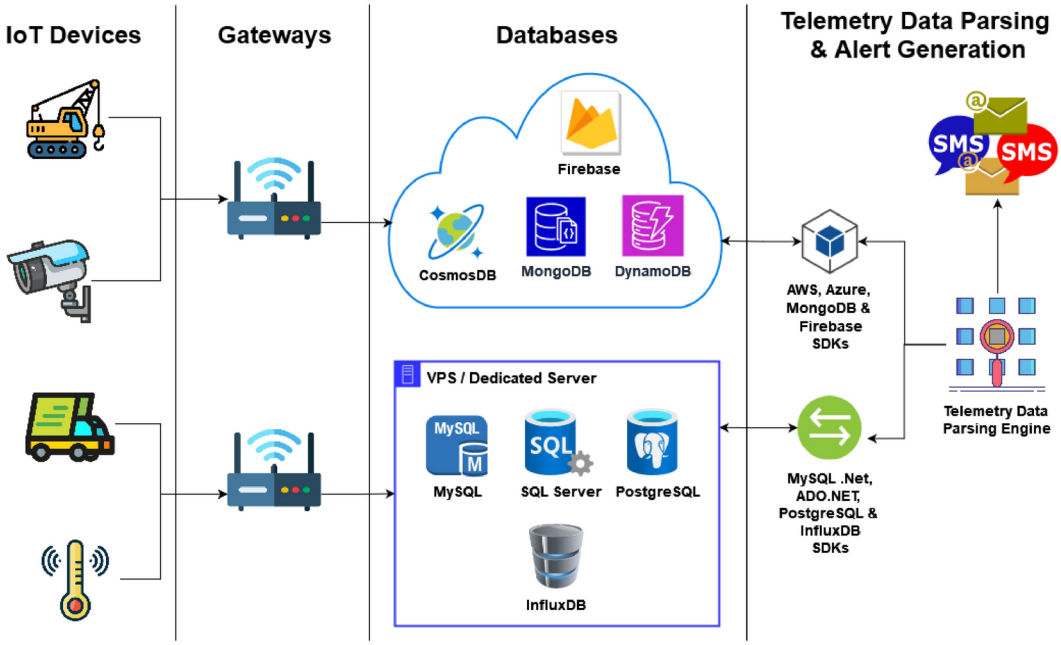


Fig. 3. End-to-end process flow.

In layer4, a diagnostic inference engine is shown which is the heart of the solution. The inference engine gets the data from databases and parses the data of each device in a solution under rules and generates alerts. The rules are designed according to the telemetry data attribute names and are associated with the recipients of diagnostic alerts.

In *layer 5*, the alerts are in the form of an SNS, SMS, or Emails to the designated/configured users for that specific type of diagnostic alert.

Proposed framework

In this study, the inference engine model was selected for its ability to efficiently handle both real-time diagnostics and predictive prognostics. The model was chosen for its flexibility in adapting to different operational conditions, along with its support for rule-based decision-making through forward and backward chaining[13]. To validate its performance, key metrics such as real-time detection accuracy, prediction reliability, and response times under varying loads were employed. These metrics were essential for assessing the system's overall effectiveness in real-world scenarios.

Moreover, the selection of hyperparameters for the diagnostic and prognostic inference engine was a critical aspect of optimizing model performance. Randomized Cross-Validation (CV) was employed for hyperparameter tuning, enabling efficient exploration of a wide range of values to select the optimal settings for each model. This method was particularly effective in addressing noisy data and improving the robustness of the system. Through iterative adjustments, key parameters such as threshold values, rule configurations, and response time settings were fine-tuned to maximize the accuracy of real-time diagnostics and future prognostics.

Prerequisites: Device telemetry data

IoT device telemetry data is paramount to information and is mandatory for this system to work. Telemetry data from various IoT devices is stored in the different databases via gateways through SSL channels via MQTT and other protocols.

Simulation environment and assumptions

The proposed framework assumes an IoT ecosystem where devices transmit telemetry data in a simulated environment replicating real-world use cases, including smart grid substations, healthcare units, houses, and farming units. The simulation environment incorporates MQTT brokers for communication, TLS-based encryption for secure channels, and cloud or on-premise database storage. Devices are simulated to generate telemetry data such as temperature, humidity, and energy consumption every 10 seconds.

Simulation tools like AWS IoT Core, Node-RED for MQTT integration, and InfluxDB with Grafana are used for telemetry visualization and monitoring. Moreover, an assumption is made that all IoT devices follow a uniform heartbeat interval for seamless integration with the diagnostic engine.

Diagnostic & prognostic engine

The diagnostic and prognostic engine serves as the heart of the system's inference engine, designed to work with both real-time and offline data based on specific requirements, though ideally tailored for real-time systems. To set up the engine, a data source, typically a database, is required, along with the necessary connector and

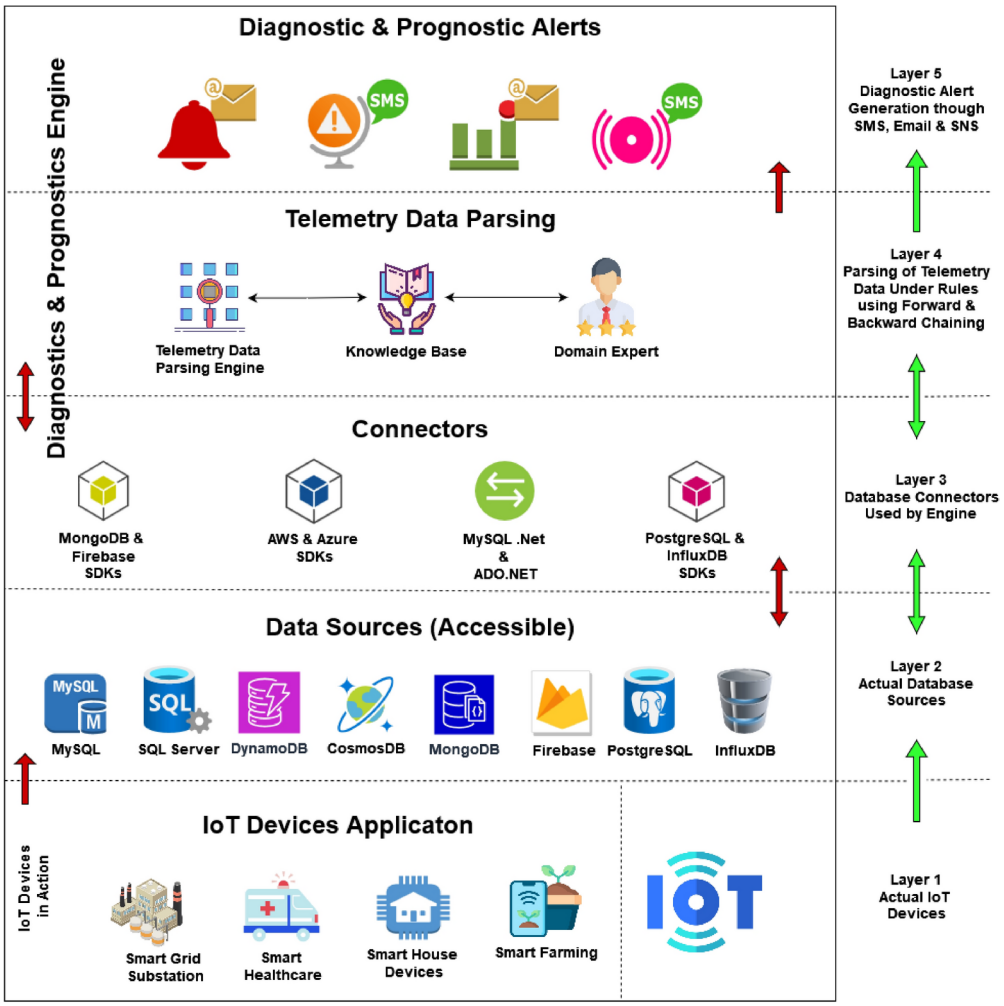


Fig. 4. Layered architecture of IoT device diagnostics & prognostics using telemetry data.

connection details. Once the database is configured, pre-defined rules can be created based on the data attributes. For example, in the simulation environment, a rule might be written to generate an alert if the temperature exceeds 70°C. In this case, the engine compares the temperature value with the threshold to determine if an alert should be triggered, with recipients for such alerts also being configurable.

The integration of these pre-defined rules represents a significant advancement in the field by providing a structured, rule-based approach to both real-time diagnostics and predictive prognostics. Leveraging predefined thresholds and patterns observed in telemetry data, the system automatically triggers alerts and generates predictions about device performance and potential failures. The combination of forward and backward chaining techniques further enhances the system's capabilities, enabling both immediate detection of issues and proactive forecasting of future risks.

Diagnostics engine

Device diagnostics using telemetry data through forward chaining is a difficult method in artificial intelligence. Telemetry data, including time measurements and device information, serves as the basis for this process. Chaining works by starting from the available data and applying rules or logic to draw conclusions or perform analytical analysis about the current or future state of the device.

In this case, the forward chaining begins with the collection and analysis of telemetry data from sensors or devices. These data may include a variety of information, including temperature readings, pressure values, error logs, operating states, or any other relevant measurements. The important thing is the continuous data entry, providing a snapshot of the device's performance and health.

As the algorithm processes incoming telemetry data, it checks the data against rules. For instance, in the simulation environment, temperature readings and error logs are monitored for anomalies in devices under predefined thresholds. Alerts or notifications are generated in cases of potential malfunctions or performance issues, enabling authorities to make better decisions.

Prognostics engine

Device prognostics using backward chaining in artificial intelligence involves prognosticating the future state or behavior of a device grounded on observed data and predefined rules. Backward chaining starts with a desired goal or thing and works backward to identify the conditions or events that would lead to that goal. This approach is particularly useful for forecasting device failures or performance declination grounded on data and known patterns.

Backward chaining involves defining rules or conditions that indicate impending failures or deteriorating performance. These rules are constructed grounded on historical data, given failure modes, and patterns associated with device malfunctions. For case, if a certain combination of sensor readings or functional parameters historically precedes a device failure, backward chaining rules can be formulated to identify these patterns.

Once the algorithm identifies the conditions or events that anticipate the awaited device failure or performance issue, it generates prognostic cautions or predictions. These alerts provide valuable insights into the likelihood and timing of future issues, enabling proactive maintenance, preemptive actions, or scheduled repairs to prevent or mitigate potential device failures. Backward chaining in device prognostics thus empowers organizations to anticipate and address issues before they escalate, optimizing device reliability, minimizing downtime, and enhancing overall operational efficiency.

Database connectors

To connect data sources with the diagnostic and prognostic engine, database connectors are used. In order to connect SQL Server databases to the inference engine, OLEDB or ADO.Net tools are employed. Similarly, MySQL Connector is used to connect the inference engine with MySQL databases, and AWS .Net SDK is used for integrating AWS DynamoDB.

Simulation tools such as MySQL Workbench and Amazon DynamoDB Local are used to simulate the database environments for testing database connectivity and query execution in a controlled setup.

Flexible rules (Knowledge Base)

In an Expert System Framework an artificial intelligence system's knowledge base, or engine, is the fundamental collection of rules, facts, and heuristics that direct the system's decision-making. The Knowledge Base, which consists of an extensive collection of organized data, represents the experience and subject-matter expertise of human experts. This framework of rules controls how the system analyzes and comprehends incoming input, giving it the ability to simulate human-like logic and problem-solving. The system can draw conclusions, make predictions, and provide suggestions thanks to these well-crafted rules that encapsulate the complexities of decision-making. Expert systems are very versatile and adaptive because of the Knowledge Base, which enables them to grow and change as new knowledge becomes available or as the system acquires more experience.

Diagnosis knowledge-Base rules (Pattern)

Diagnostic knowledge base rules serve as a standardized method for evaluating telemetry data and generating diagnostic evaluations based on pre-defined criteria in the rules. These rules are designed to analyze specific conditions or parameters within the telemetry data, helping to identify errors or deviations from expected behavior. For example, considering criteria such as RxRSSI values falling below certain thresholds or RxLQI metrics indicating low quality, these rules provide standardized criteria for defining issues based on different communication systems, such as ISM or Cellular as shown in Figure 5 and Figure 6.

These rules guide the diagnostic process, allowing the system to identify specific fault definitions and possible solutions. Recommended solutions, such as adjusting transmission power or channel configuration decisions, are defined in knowledge-based rules to help solve these problems effectively. These rules allow the system to analyze, organize, and support actions based on the deviations observed in the telemetry data, thus enabling quick maintenance and optimal device operation. There are a couple of rules shown in Figure 7 and Figure 8 regarding humidity and PCB temperature.

The threshold values can be changed based on the requirements and experience with the device behaviors. Following is the sample XML-based diagnostic rule format as shown in Figure 9.

Another way of writing rule is shown in Figure 10, this approach is shown in code form but is being populated from the database instead.

Both XML and DB-based rules can be written and used because both are flexible and easy to modify. There is an important concept of parsing the data under rules for a specific device. For example, in the case of RX-RSSI, how many consecutive records need to be processed and analyzed that the value is below the threshold to generate an alarm/alert? The proposed solution has the capability to process a certain number of records before generating alarms/alerts.

Parsing of telemetry data under rules

The process of parsing telemetry data under rules is fundamental to the operation of an inference engine and is essential for understanding incoming data and triggering follow-up actions. Parsing of telemetry data under rules can be done in two ways:

- Forward Chaining,
- Backward ChainingThe system implements a forward chaining mechanism in which the engine systematically loops through stored facts and rules to derive new information. Navigate through telemetry data and compare it to predefined conditions set in rules. For example, this algorithm tracks consecutive records that meet the conditions of a particular rule. When a specified number of consecutive records match the condition, a

Criterion

- RxRSSI < -60
- RxLQI < 50

Rules

- **ISM**
  - "RxRSSI" is less than threshold
  - "RxLQI" is less than threshold
- **Cellular**
  - "CellularModemRSSI" is less than threshold
  - "CurrentRFOutput" is less than threshold

Fault Description and Remedy (RxRSSI)

- RxRSSI depicts signal strength at receiver end.
- Decrement in its value below threshold limits means that communication taking place between transmitter and receiver is no more reliable since received signal is very weak as compared to channel noise
- To help leverage your signal strength measurement most effectively so you can make channel planning decisions. Increasing transmitter power could also help to resolve this issue

Diagnosis

- RxRSSI Value has decreased below threshold values

Fig. 5. RF communication signal strength (RxRSSI, Time) - low-quality communication.

Criterion: RxLQI is less than its threshold value.
Sensors and Variable involved:

- Rx-LQI
- Time/Date Stamp

Diagnosis: Low quality communication due to decrement in RxLQI beyond its lower threshold value.
Description: RxLQI is measure of quality of received signal. This tells us how easily a received signal can be demodulated. If LQI value decreases below threshold value then data packet will be dropped.
Recommendations: Use transmitter and receiver with higher efficiency, since LQI depends upon environmental factors and electrical noise of transmitter and receiver.

Fig. 6. RF communication signal strength (RxLQI, Time)- low-quality communication.

new fact or alarm is generated and an action is triggered based on the defined outcome. This iterative process allows you to identify patterns and correlations within your data stream. This is important for proactive decision-making based on evolving data trends.

Conversely, the backward chaining method starts with a desired result or outcome and evaluates existing facts against rules to determine the necessary conditions. If the conditions match facts stored in the system, a new fact or alarm is generated indicating that the desired result has been achieved. This approach is particularly useful

Criterion: If enclosure humidity is greater than its threshold value AND this condition persists for a minimum threshold time then PLG enclosure humidity is too high.
Sensors and Variable involved:

- Humidity
- Time/Date Stamp

Diagnosis: Increase in enclosure humidity. Possible leak in enclosure.
Description: Electrical circuits are designed to operate in specific environmental conditions. Humidity specification is also among those conditions. Increase in humidity beyond threshold value will cause sensors degradation.
Recommendations: Check humidity level of surrounding area.

Fig. 7. Enclosure humidity (Humidity, Time) - elevated humidity.

Criterion: If enclosure humidity is increased AND PCB temperature greater than its threshold value for a specific minimum time then high severity warning for PCB issue.
Sensors and Variable involved:

- PCB Temp Sensor
- Time/Date Stamp.

Diagnosis: PCB at risk of Failure.
Description: PCB temperature remains within a permissible range. If it exceeds its threshold values then it is caused by some failure at PCB.
Recommendations: Check behavior of device by monitoring telemetry for particular alarm generating unit.

Fig. 8. High severity PCB temperature (PCB Temp, Time) - elevated PCB temperature.

when the system requires certain conditions to be met for a particular outcome or action, and the data can be inspected to determine the existence of the required conditions.

Diagnostic alert type and recipients

The following code 3 section, in Figure 11, shows a sample diagnostic alert that is being generated due to an Rx-RSSI value less than the threshold for more than 5 consecutive records. The design of the alert is dynamic and in HTML format with placeholders to be replaced with real-time values. For example, the subject, recipients, and body of the alert is dynamic. Alerts' recipients can be managed and configured alert-wise in the diagnostics engine so that proper intimation can be delivered to the desired person in time through an interactive graphical interface.

Results and discussion

Experimental design

The experimental design of this research aims to improve the field of device diagnostics and prognostics using telemetry data and both forward and backward chaining algorithms. The main goal is to develop a robust framework for detecting patterns and anomalies in telemetry data that can indicate potential issues with devices connected to the IoT. Forward chaining is employed for real-time issue detection, while backward chaining is utilized for predictive insights and future risk assessment. The most important goal is to improve the reliability

```
<xml version="1.0" encoding="UTF-8">
  <Diagnostics>
    <DiagnosticType id="1" desc="Low Quality Communication Problem" enabled="false" version="1">
      <Criteria>
        <Criterion id="1" desc="RX-RSSI < -60" index="1">
          <Rules desc="" RulesCount="1">
            <Rule id="1" desc="" cf="100" severity="High">
             <ConsectiveRecordsToBeProcessed> 5 </ ConsectiveRecordsToBeProcessed >
             <Diagnosis> "RXRSSI is less than threshold" </Diagnosis>
             <Action>
               <Email enabled="true">
                 <subject> "Low Quality Communication Alert"</subject>
                 <Body> "Low Quality Communication Problem" </Body>
                 <From> abc@diagnosticengine.com </From>
                 <Recipient>
                   <To enabled="true">
                     <EmailAddress enabled="true"> test@test.com </EmailAddress>
                   </To>
                   <CC enabled="true"> </CC>
                 </Recipient>
               </Email>
             </Action>
            </Rule>
          </Rules>
        </Criterion>
      </Criteria>
    </DiagnosticType>
  </Diagnostics>
</xml>
```

Fig. 9. Diagnostic XML-based rule.

```
List<Fact> conditions = new List<Fact>
        {
            new Fact("Temperature", "High")
        };
Fact consequence = new Fact("Alert: High Temperature",
"1");
Rule rule = new Rule("Rule 1", conditions, consequence,
2); // Check for 2 consecutive records
inferenceEngine.AddRule(rule);
```

Fig. 10. Diagnostic DB or code-based rule.

and efficiency of the inspection process, making the pressure faster and enabling the faster resolution of emerging device issues.

Research questions and objectives

The research questions underlying this experiment focused on the effectiveness of advanced artificial intelligence algorithms in analyzing telemetry data for early detection of devices using forward and backward chaining. Specific goals include:

- To check the device diagnostics using the forward chaining directly to identify patterns that indicate possible device failure.
- Check the device prognostics using backward chaining to prognosticate the future state or behavior of a device grounded on observed data and predefined rules.
- Quick review of telemetry data analysis for quick problem identification.

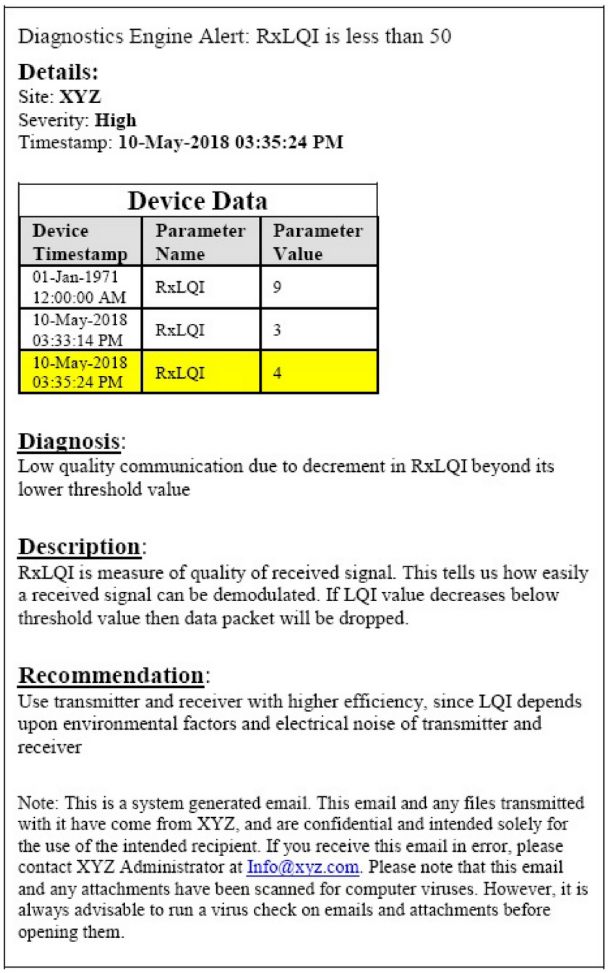
Materials and tools

These experiments are conducted using various IoT devices equipped with sensors to measure parameters such as temperature, pressure, and signal strength. The telemetry data produced by these devices is the main input to the forward and backward chaining algorithms (Inference Engine). The test setup includes various tools as follows:

- **Data Connectors:** MySQL, ADO.NET, DynmoDB SDK, Microsoft Office Interop Excel
- **Database:** MySQL 5.6, SQL Server Management Studio 2022 and Notepad (CSV)
- **Programming Language:** C#

Procedure

The Telemetry data from IoT devices placed in a controlled environment displaying different operating conditions is collected in real-time or offline and stored in a database. Forward and backward filtering algorithms are applied to analyze the telemetry data. The algorithm is designed to detect anomalies using current telemetry



Diagnostics Engine Alert: RxLQI is less than 50

Details:
Site: **XYZ**
Severity: **High**
Timestamp: **10-May-2018 03:35:24 PM**

Device Data

Device Timestamp	Parameter Name	Parameter Value
01-Jan-1971 12:00:00 AM	RxLQI	9
10-May-2018 03:33:14 PM	RxLQI	3
10-May-2018 03:35:24 PM	RxLQI	4

Diagnosis:
Low quality communication due to decrement in RxLQI beyond its lower threshold value

Description:
RxLQI is measure of quality of received signal. This tells us how easily a received signal can be demodulated. If LQI value decreases below threshold value then data packet will be dropped.

Recommendation:
Use transmitter and receiver with higher efficiency, since LQI depends upon environmental factors and electrical noise of transmitter and receiver

Note: This is a system generated email. This email and any files transmitted with it have come from XYZ, and are confidential and intended solely for the use of the intended recipient. If you receive this email in error, please contact XYZ Administrator at Info@xyz.com. Please note that this email and any attachments have been scanned for computer viruses. However, it is always advisable to run a virus check on emails and attachments before opening them.

Fig. 11. Sample diagnostic alert/alarm.

values and historical trends in the designed rules. Diagnostic tests are performed based on the detected patterns, allowing immediate action and response to potential device problems. Prognostic assessments are made based on goals to drill down to the facts that may cause issues in the future.

Data collection

Telemetry data is regularly collected from IoT devices using a secure communication channel, considering factors such as RxRSSI (Signal Quality), Temperature, Humidity, and other relevant metrics. The data collection process enables a complete monitoring of the behavior of the device in different situations, contributing to the reliability of the research. This study used actual telemetry data provided by Dr. Nour Moustafa in his research collected from IoT and IIoT sensors with various operating systems[14]. The following are the two datasets, used in this study:

- IoT Normal Fridge[15],
- IoT Normal Weather[15]

Experimental setup

Data

IoT data (in the form of CSV and Excel files) is used from the mentioned source and treated that as real-time data (can be considered offline as well) as we keep the record datetime in consideration while parsing data under rules and always parse new data only.

Computer

The computer with the following specifications is used for experiments:

- Generation 10th
- Processor i7
- Brand HP
- RAM 16GB

```
List<Fact> conditions = new List<Fact>
        {
            new Fact("Temp_Condition", "High")
        };
Fact consequence = new Fact("Alert: High
Temperature", "1");
        Rule rule = new Rule("Rule 1",
conditions, consequence, 10); // Check for 10
consecutive records
        inferenceEngine.AddRule(rule);
```

Fig. 12. Rule for high-temperature alarm.

```
List<Fact> conditions1 = new List<Fact>
        {
            new Fact("humidity", "32.0")
        };
Fact consequence1 = new Fact("Alert: Low
Humidity", "1");
        Rule rule1 = new Rule("Rule 2",
conditions1, consequence1, 5); // Check for 5
consecutive records
        inferenceEngine.AddRule(rule1);
```

Fig. 13. Rule for humidity alarm.

Fact Generated	Type	Condition
Alert: High Temperature	Diagnostic	Detected in real-time
Alert: High Temperature	Diagnostic	Detected in real-time
Alert: High Temperature	Diagnostic	Detected in real-time
Alert: High Temperature	Diagnostic	Detected in real-time
Alert: Low Pressure = 32	Diagnostic	Detected in real-time
Alert: Low Pressure = 32	Diagnostic	Detected in real-time
Alert: Low Pressure = 32	Diagnostic	Detected in real-time
Alert: Low Pressure = 32	Diagnostic	Detected in real-time

Table 1. Diagnostics and prognostics engine console diagnostic.

- ROM 250 SSD

Software setup
The diagnostics and prognostics engine (Inference Engine) is installed on the above-mentioned machine and performed the configuration regarding the data file path and others related to the access rights. Designed following rules as shown in Figures 12 and 13.

The rule in Figure 12 worked on the IoT Normal Fridge as there is no humidity-related field in this dataset. The algorithm generated few alarms, as consecutive 10 records with high temperatures were required to generate the alarm. The rule in Figure 13 worked on the IoT Normal Weather dataset and generated a lot of alarms at different times as the threshold was according to the required threshold and consecutive records to generate alarms were also 5.

Performance metric
The performance of the diagnostic and prognostic systems was evaluated using several key metrics. For diagnostics, real-time detection of alerts, such as high temperature and low pressure, demonstrated the system's ability to accurately identify critical conditions. The diagnostic engine consistently detected these issues in real time, ensuring timely responses to system anomalies. In terms of prognostics, the system not only predicted future faults, such as high temperature and low pressure, but also derived additional insights based on current and future data, highlighting its ability to forecast potential device failures.

Results

The results presented in the diagnostics and prognostics engine console (Table 1) reflect real-time detections of specific system conditions, with a focus on temperature and pressure alerts. The repeated appearance of "Alert: High Temperature" and "Alert: Low Pressure = 32" signifies recurring issues identified by the system, indicating either persistent system abnormalities or the presence of consistent operating conditions that may

Fact Generated	Type	Condition
Alert: High Temperature	Prognostic	Predicted for future data
Alert: Low Pressure = 32	Prognostic	Predicted for future data
Temperature = High	Prognostic	Derived from current/future data
Pressure = Low	Prognostic	Derived from current/future data

Table 2. Diagnostics and prognostics engine console prognostic.

require attention. These observations point to challenges in maintaining stable operating conditions, particularly in environments prone to fluctuating temperature and pressure parameters, which are critical for IoT systems. Specifically, the fact that these alerts are categorized as "Diagnostic" and marked as "Detected in real-time" emphasizes the engine's immediate responsiveness to critical parameters such as temperature and pressure. The repetition of these alerts in real time suggests that the system is actively monitoring and flagging conditions that exceed predefined thresholds, which could be indicative of potential operational failures or the need for maintenance. However, these repeated alerts also highlight a challenge in distinguishing between temporary anomalies and chronic issues, necessitating advanced filtering and prioritization mechanisms to avoid false positives or alert fatigue. The frequent occurrence of similar alerts, particularly in a real-time context, highlights the engine's capability to identify ongoing concerns, allowing for timely intervention and troubleshooting.

The results from the diagnostics and prognostics engine console (Table 2), presented as prognostic facts, highlight predictions based on current and future data. The alerts for "High Temperature" and "Low Pressure = 32," marked as "Predicted for future data," suggest that the system is forecasting potential issues before they occur, which is a key feature of predictive maintenance. The system is proactively identifying possible future problems with temperature and pressure, thus enabling preventive actions. Additionally, the derived facts, "Temperature = High" and "Pressure = Low", based on both current and future data, imply that the engine is not only evaluating real-time conditions but is also considering trends or patterns that might indicate an impending failure or abnormal performance. This predictive capability addresses one of the major challenges in the field, anticipating potential failures with sufficient lead time to prevent operational disruptions. However, the accuracy of these predictions depends heavily on the quality and completeness of telemetry data, which remains a significant hurdle in the deployment of reliable prognostic systems. This predictive capability is essential for long-term system reliability, as it allows for preemptive measures such as maintenance scheduling or system adjustments, ultimately reducing downtime and improving overall operational efficiency.

Interpretation and discussion

Elucidating the results obtained from the application of the inference engine and device diagnosis and prognostics based on telemetry data provides a very important insight into the effectiveness and capabilities of this method in real-world situations. It is pertinent to mention that the proposed approach integrates both diagnostic and prognostic capabilities, enabling real-time issue detection and predictive insights into future system performance. Unlike traditional methods that focus on either diagnostics or prognostics, the proposed dual approach offers comprehensive monitoring and proactive management of IoT devices. Rule-based inference and adaptive thresholds help address current failures while anticipating potential risks, optimizing maintenance schedules, and reducing downtime. This discussion focuses on the implications of the results, their implications for device maintenance plans, and the limitations encountered during the experiment.

Accuracy and speed of diagnostics

Applications of forward-chaining algorithms have shown significant potential in detecting patterns and potential anomalies in telemetry data streams. This method is effective in quickly identifying the wrong application and behavior of the device, especially in the case of situations where the defined rules are associated with the following problems. The continuous chain process demonstrates the accuracy of the diagnostic tests, enabling early detection and intervention to reduce potential device errors.

Effectiveness of backward chaining

On the other hand, the implementation of the above chain has shown its value in removing the potential causes of the identified problems. The backward chaining method or the prognostics part of the inference engine has been shown to help in tracing levels or outcomes back to their original conditions and help in identifying the prediction of possible device failures. However, its effectiveness depends on the completeness and correctness of the rules that are applied, thus revealing the natural dependence on the system of pre-defined rules for a fair review.

Observations and inferences

The results of the diagnostic and prognostic systems reveal several key observations. The diagnostic engine's ability to detect real-time alerts for conditions like high temperature and low pressure indicates that the system is effective in providing immediate responses to device anomalies. Meanwhile, the prognostic engine's predictive capabilities highlight its strength in anticipating future issues based on historical and real-time data. This combination of real-time diagnostics and predictive forecasting suggests that the proposed system not only ensures immediate corrective actions but also allows for proactive maintenance, reducing the risk of unplanned

downtime. The system's performance thus reinforces its potential for enhancing device reliability and operational efficiency across diverse IoT environments.

Practical implications for IoT maintenance
The results of this study highlight the practical benefits of outsourcing inference engines for device analysis in IoT environments. The ability to predict and detect device malfunctions using telemetric data offers promising changes in predictive maintenance strategies. The combination of forward and backward integration methods provides a comprehensive analysis process that not only defines problems but also defines their causal relationships, making the maintenance work more efficient. It works to improve device reliability and reduce downtime.

Scope and future direction
Despite the promising results, there are several limitations during this study. The effectiveness of inference engines depends on the quality and completeness of the pre-defined rules, which shows the need to keep refining and adapting the rules in different areas of work. In addition, scalability and computational efficiency are still major challenges, allowing further research into the safe implementation of these techniques in large-scale IoT deployments. Moreover, direct data access to an inference engine for timely alarms can greatly help the authorities in making better decisions and timely actions.

Conclusions and future work
Conclusion
The research presented here demonstrates the effectiveness and potential of using device diagnostics and prognostics inference engines for device detection based on telemetry data in IoT systems. The study demonstrated the practical application of forward and backward chaining algorithms to block and solve device problems. The results show the importance of applying a rules-based approach to the monitoring process effectively, helping to speed up and improve device reliability in the IoT environment. The successful application of diagnostics and prognostics inference engines in detecting device issues based on telemetry data also lays the foundation for better predictive maintenance practices. This process provides valuable insight into potential faults, and provides a continuous approach to device maintenance, thereby reducing downtime and improving performance in IoT infrastructure.

Future work
Several avenues for future research emerge from this study, providing opportunities to improve and expand the scope of device diagnostics and prognostics inference engines in IoT device discovery.

- An updated rule-based system: Continuous redesign and adaptation of rules to adapt to a complex work environment is critical. Future work includes the development of a flexible, self-learning process that starts with the device behavior change.
- Scalability and Computational Performance: It is still important to solve problems related to the scalability and computational efficiency of inference engines. Research focused on improving algorithms and parallel processing systems will enable seamless implementation and large-scale IoT deployment.
- Integrating machine learning: Integrating machine learning algorithms with rule-based inference systems can improve detection. Hybrid model studies combining the description of the system based on the predictive power of machine learning are promising for more robust research.
- Real-time predictive analytics: Advances in real-time predictive analytics for real-time anomaly detection and automated decision-making will greatly benefit the maintenance of IoT devices. Future studies could investigate real-time screening techniques to speed up response times in critical situations.
- Conduct further experiments under diverse environmental conditions, using a broader range of data sources and operational settings, to evaluate system performance and refine accuracy. Additionally, explore integrating more diagnostic and prognostic rules, along with experimenting with different thresholds, to enhance the method's robustness and scalability across various use cases.In conclusion, the adoption of device diagnostics and prognostics inference engines for device fault detection based on telemetry data represents a significant step for proactive management strategies in the IoT environment. Acknowledging these findings and pursuing future research avenues will strengthen the role of diagnostics and prognostics inference engines in ensuring the reliability and longevity of IoT devices.

Data availibility
The data can be requested from the corresponding authors.

Code availability
Not applicable.

Received: 10 October 2024; Accepted: 4 February 2025
Published online: 04 March 2025

References
1. MacGillivray, C. & Reinsel, D. Worldwide Global Data Sphere IoT Device and Data Forecast, 2020–2024. IDC; (2020).

2. Al-Ajlan, A. The comparison between forward and backward chaining. *International Journal of Machine Learning and Computing.* **5**(2), 106 (2015).
3. Waterman, D.A. & Hayes-Roth, F. An overview of pattern-directed inference systems. Pattern-directed inference systems. p. 3–22 (1978).
4. Pandey, S. Industry 5. 0, An Idea of Smart Human Centric Industrial Revolution.;.
5. Hassanien, A. E., Darwish, A. & Abdelghafar, S. Machine learning in telemetry data mining of space mission: basics, challenging and future directions. *Artificial Intelligence Review.* **53**(5), 3201–3230 (2020).
6. Ayvaz, S. & Alpay, K. Predictive maintenance system for production lines in manufacturing: A machine learning approach using IoT data in real-time. *Expert Systems with Applications.* **173**, 114598 (2021).
7. Zhou, C., Guo, D. & Zhang, C. Research on Server Fault Diagnosis Based on Expert System. In: Advanced Manufacturing and Automation X 10. Springer; p. 213–220 (2021).
8. Saiful, M. & Nur, A.M. Application of Expert System with Web-Based Forward Chaining Method in Diagnosing Corn Plant Disease. In: Journal of Physics: Conference Series. vol. 1539. IOP Publishing; p. 012019 (2020).
9. Wang, C., Lu, N., Cheng, Y. & Jiang, B. A telemetry data based diagnostic health monitoring strategy for in-orbit spacecrafts with component degradation. *Advances in Mechanical Engineering.* **11**(4), 1687814019839599 (2019).
10. Anwar, M. R. Analysis of expert system implementation in computer damage diagnosis with forward chaining method. *International Transactions on Artificial Intelligence.* **1**(2), 139–155 (2023).
11. Sirait, H. A Basic Elements and Characteristics in Building an Expert System. *Journal Neosantara Hybrid Learning.* **1**(3), 227–248 (2023).
12. Morchid, A. et al. IoT-based smart irrigation management system to enhance agricultural water security using embedded systems, telemetry data, and cloud computing. *Results in Engineering.* **23**, 102829 (2024).
13. Yan, W., Wang, J., Lu, S., Zhou, M. & Peng, X. A review of real-time fault diagnosis methods for industrial smart manufacturing. *Processes.* **11**(2), 369 (2023).
14. Moustafa, N. & Slay, J. UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In,. military communications and information systems conference (MilCIS). *IEEE* **2015**, 1–6 (2015).
15. Moustafa, N. A new distributed architecture for evaluating AI-based security systems at the edge: Network TON_IoT datasets. *Sustainable Cities and Society.* **72**, 102994 (2021).

Author contributions
MSF conceptualization, data curation, writing-original manuscript. RPM conceptualization, formal analysis, writing-original manuscript. AA methodology, formal analysis, data curation. KT software, methodology, project administration. EGV investigation, funding acquisition, visualization. FA validation, visualization, investigation. HK visualization, software, resources. I.A. supervision, validation, writing-review & edit manuscript. All authors reviewed the manuscript and approved it.

Funding
This research was funded by the European University of Atlantic. This research was also funded by the Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2025R77), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

Declarations

Conflict of interest
The authors declare no conflict of interests.

Ethics approval
Not applicable.

Consent to participate
Not applicable.

Consent for publication
Not applicable.

Additional information
Correspondence and requests for materials should be addressed to F.A. or I.A.