



上海交通大学
SHANGHAI JIAO TONG UNIVERSITY



The Institution of
Engineering and Technology

ORIGINAL RESEARCH OPEN ACCESS

Intelligent Advisor System for Prescriptive Maintenance of Engineered Assets Using Failure Modes, Effects and Criticality Analysis, Knowledge Graph and Machine Learning

Hongyi Lin | Agusmian Partogi Ompusunggu

Cranfield University, Faculty of Engineering and Applied Sciences (FEAS), Bedfordshire, UK

Correspondence: Agusmian Partogi Ompusunggu (agusmian.ompusunggu@cranfield.ac.uk)

Received: 1 February 2026 | **Revised:** 2 May 2026 | **Accepted:** 8 May 2026

Keywords: chatbot | fault diagnosis | FMECA | intelligent question answering (Q&A) | knowledge graph (KG)

ABSTRACT

Maintenance strategies for engineered assets are shaped by the diversity of potential failure modes and their impact on functional performance and safety. As these assets comprise numerous interconnected components, they are prone to concurrent degradation and multi-source faults. Consequently, maintenance personnel face challenges in efficiently and accurately identifying specific failure types and pinpointing root causes to plan effective interventions. This paper introduces an intelligent advisor—a question answering (QA) system—for fault diagnosis, failure mode identification and automated root-cause analysis by integrating failure modes, effects and criticality analysis (FMECA), machine learning (ML) and a knowledge graph (KG). A linear actuator serves as a case study to validate the approach via a four-stage implementation: (i) FMECA construction from design knowledge and known degradation mechanisms; (ii) signal processing and feature engineering from sensor data collected under multiple fault scenarios; (iii) digitisation of FMECA into a KG to represent concepts and relations; and (iv) a natural language processing (NLP) layer that enables user-friendly interaction. Results show that the integrated framework enhances the interpretability and traceability of automated diagnostics, providing a transparent pathway from sensor-level anomalies through KG-based reasoning to prescriptive maintenance actions.

1 | Introduction

1.1 | Background

With the emergence of Industry 4.0, intelligent management and production technologies have become central to modern industrial manufacturing. Within this context, predictive maintenance (PdM) has been widely recognised as a key enabling technology, as it reduces equipment downtime and

maintenance costs while improving productivity and return on investment [1].

Consequently, substantial research efforts have focused on developing technologies to support the industrial adoption of PdM strategies, including sensing infrastructures, data connectivity and algorithms for fault detection, diagnosis and prognosis. For example, Ompusunggu et al. [2] developed Kalman

Abbreviations: AC, Aho-Corasick; AI, artificial intelligence; CM, condition monitoring; FMECA, failure modes, effects, and criticality analysis; GUI, graphical user interface; KG, knowledge graph; LLM, large language model; ML, machine learning; NLP, natural language processing; PCA, Principal Component Analysis; PdM, predictive maintenance; Q&A, question answering; RF, random forest; RPN, Risk Priority Number; RxM, prescriptive maintenance; SOD, severity-occurrence-detection; WO, work order.

This is an open access article under the terms of the [Creative Commons Attribution](https://creativecommons.org/licenses/by/4.0/) License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited.

© 2026 The Author(s). *Artificial Intelligence for Engineering* published by John Wiley & Sons Ltd on behalf of Institution of Engineering and Technology.

filtering-based prognostic solutions for automatic transmission clutches. Wei et al. [3] proposed a closed-loop manufacturing scheduling framework with integrated PdM capabilities, using a convolutional neural network (CNN) trained on features extracted from historical operational data. Maulana et al. [4] introduced prognostic approaches combining machine learning (ML) techniques with Bayesian filtering for aircraft engine health management.

In addition, You and Meng [5] formulated a PdM scheduling model and investigated optimal maintenance planning through numerical analysis and dynamic updates. Tod et al. [6] applied physics-based modelling to extract condition indicators from sensor data for PdM of solenoid valves, whereas Brijder et al. [7] and Vásquez et al. [8] developed PdM algorithms and software tools to address corrosion-related degradation in offshore wind turbine structures. Furthermore, Aremu et al. [9] examined health indicator (HI) staging in PdM, characterising asset life-cycle states using discrete health categories.

In contemporary PdM practice, as illustrated in Figure 1, condition monitoring (CM) systems and/or supervisory control and data acquisition (SCADA) systems collect sensor signals and operational log data. These data sources capture events generated by various applications running on industrial assets and are transmitted to a central database for fault diagnosis, prognosis and trend analysis using data-driven algorithms. When calculated condition indicators or remaining useful life (RUL) estimates exceed predefined thresholds, work orders (WOs) are automatically generated by a computerised maintenance management system (CMMS) or manually issued by a maintenance planner. The WO initiates corrective actions by maintenance personnel. In addition to sensor data, service data—such as performed maintenance activities, involved assets or tools,

consumed spare parts and observed failure types—constitute another essential information source [10]. Collectively, modern PdM strategies contribute to more standardised, digitalised and traceable maintenance processes.

Despite notable progress, current PdM implementations still exhibit several limitations. First, they often fail to provide stakeholders with a comprehensive understanding of failure root causes, resulting in limited traceability. Second, they lack mechanisms for pushing explicit fault knowledge derived from data-driven reasoning, leaving maintenance stakeholders with an insufficiently intuitive understanding of the relationship between sensor-based condition indicators and fault phenomena. Addressing these shortcomings requires a transition from PdM to prescriptive maintenance (RxM). Such a transition necessitates advisory systems—specifically, question–answering (QA) systems with root-cause reasoning capabilities—that support maintenance teams by explicitly linking fault symptoms, failure mechanisms and optimal maintenance actions, thereby minimising downtime and preventing catastrophic asset failures.

1.2 | Related Work

Failure modes, effects and criticality analysis (FMECA) is a well-established technique for identifying critical components in engineered assets and defining maintenance priorities, providing a structured foundation for maintenance planning [12]. Conventionally, FMECA information is organised in tabular form [13]. Although effective, this representation becomes increasingly difficult to interpret as asset complexity grows.

Knowledge graphs (KGs) offer an alternative paradigm for modelling complex relationships by representing real-world entities and concepts as directed labelled graphs. As illustrated

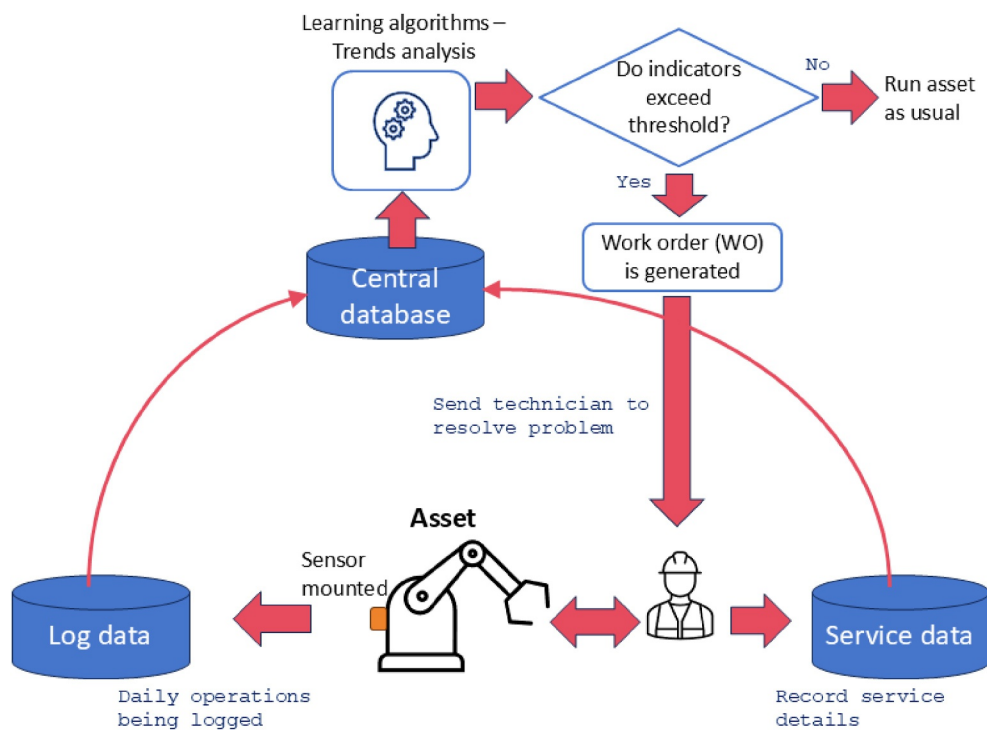


FIGURE 1 | Typical PdM workflow. Source: Adapted from Refs. [10, 11].

in Figure 2, a KG consists of four elements [14]: (i) nodes, (ii) edges connecting nodes, (iii) labels assigned to edges and (iv) an assignment function linking edges to labels. In the figure, nodes A and C represent entities that can be persons/maintenance personnel, places/sites, objects such as engineered assets, events, models, abstract concepts etc. One can view the assignment of label B to the edge E = (A and C) as a triple (A, B and C), where A, B and C represent the subject, predicate (verb) and object, respectively. The ability of KGs to efficiently encode structured domain knowledge makes them particularly suitable for representing FMECA data as interconnected triples (e.g., fault type → manifestation → failure mechanism).

Compared with tabular representations, FMECA-KGs enable flexible querying and reasoning using languages such as Cypher, which was originally developed for the Neo4j graph database [15]. Existing studies demonstrate the feasibility of modelling FMECA using KGs. For instance, Chen et al. [16] constructed a Neo4j-based FMECA-KG for power plant systems and reported over a 60% improvement in analysis efficiency. However, KG construction remains highly dependent on domain experts, making it labour-intensive and time-consuming. Similarly, Zhang et al. [17] showed that FMECA-KGs can enhance engineers' understanding of root causes, yet their approach is constrained by the availability of historical FMECA records and failure data.

In parallel, machine learning (ML) techniques have become dominant in PdM by learning failure patterns from high-dimensional sensor data [18, 19]. Methods typically rely either on feature engineering combined with classical ML algorithms (e.g., random forests and support vector machines) or on deep neural networks (NNs) that process raw sensor signals for fault diagnosis and RUL estimation [20]. Liu et al. [21] proposed a self-supervised deep learning framework for PdM under limited label availability, whereas Chen et al. [22] introduced a Cox proportional hazards deep learning model for sparse data scenarios. Nevertheless, these methods remain vulnerable to data distribution shifts and often lack interpretability, particularly for root-cause analysis. QA systems have also gained attention in maintenance applications by enabling intuitive access to maintenance knowledge through natural language processing (NLP). When integrated with KGs, QA systems can support precise and efficient PdM. Chatterjee and Dethlefs [23] developed a KG-based QA system for wind turbine maintenance, employing an encoder-decoder NLP architecture to generate Cypher queries for automated knowledge retrieval.

More recently, researchers have explored the integration of QA systems, KGs and large language models (LLMs). Kohl et al. [24] proposed an intelligent dialogue system that combines vector

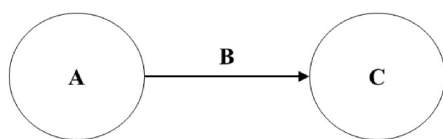


FIGURE 2 | A triple in a directed labelled graph. *Source:* Adapted from Ref. [14].

databases and KGs within a retrieval-augmented generation framework, reducing LLMs hallucinations and supporting multi-agent reasoning for maintenance planning. Other studies have introduced LLMs into maintenance assistants to improve natural language understanding and decision support [25, 26]. However, these approaches primarily focus on conversational capabilities and user interaction rather than rigorous automated root-cause analysis. Moreover, LLMs may face constraints such as limited device resources and frequent model evolution, which restrict their inference efficiency [27]. In specialised industrial scenarios characterised by limited sensor data and sparse textual corpora, LLMs-based solutions often underperform.

1.3 | Research Gap and Aim

Advanced question-answering (QA) systems and conversational assistants for prescriptive maintenance constitute a rapidly growing research area. Although substantial progress has been achieved in machine learning (ML)-based fault classification, failure modes, effects, and criticality analysis (FMECA), knowledge graph (KG) modelling and QA system architectures, these research streams have largely evolved in isolation. Consequently, their systematic integration into a unified QA framework for prescriptive maintenance (RxM)—capable of automated fault root-cause analysis and actionable maintenance recommendation—remains limited and fragmented. Furthermore, the feasibility of such integrated solutions under conditions of small text corpora and limited sensor data, which are common in real-world industrial environments, has not yet been convincingly demonstrated.

Existing efforts to combine FMECA and KGs have primarily focused on representing failure knowledge and facilitating interaction between users and knowledge bases. For instance, studies integrating failure mode analysis with KGs [28] mainly emphasise the system's ability to recognise entities and relationships in natural language queries and map them onto graph structures. In parallel, large language models (LLMs) have gradually been incorporated into maintenance assistants, as demonstrated by Yang et al. [25] and Cao et al. [26]. However, these approaches predominantly centre on enhancing natural language understanding, dialogue generation and decision-support interaction, rather than on establishing a rigorous and explainable linkage between data-driven fault prediction results and structured maintenance knowledge. To address these gaps, this paper proposes a novel QA system for automated maintenance prescriptions that integrates ML-based fault prediction with a knowledge graph constructed from FMECA data (hereafter referred to as the FMECA-KG). A rule-based and template-driven question-response mechanism is adopted to map fault prediction outputs onto the FMECA-KG, enabling explicit reasoning about failure root causes and corresponding maintenance actions. Through either user queries or experimental data input, stakeholders can directly obtain prescription results, including predicted faults, associated root causes encoded in the FMECA-KG and recommended maintenance interventions. A linear actuator system is employed as a case study to demonstrate the feasibility and effectiveness of the proposed integrated framework under realistic data constraints.

1.4 | Scope and Contributions

The scope of this paper is confined to an integrated framework comprising ML-based data prediction using a random forest model, FMECA-KG construction and a rule-based question-answering (QA) system. As discussed in Section 1.3, existing applications of pre-trained language models and large language models (LLMs) in maintenance systems predominantly focus on system-level natural language understanding, entity recognition and semantic mapping between user queries and knowledge bases. In contrast, the emphasis of this study lies in establishing a rigorous and explicit linkage between experimental sensor data and structured maintenance knowledge, as represented by an FMECA-KG.

By adopting a rule-based natural language processing strategy, this research fixes the semantic mapping between fault prediction outcomes and KG entities through predefined and domain-consistent patterns. This design choice prioritises robustness, interpretability and predictability over linguistic flexibility. More importantly, it enables deep integration between the ML model's data-driven understanding of real-world industrial equipment behaviour and structured expert knowledge embedded in the FMECA-KG. In doing so, this work validates and operationalises an intelligent maintenance paradigm that progresses systematically from 'dataset $\rightarrow$ analysis $\rightarrow$ understanding $\rightarrow$ knowledge'.

The key contributions of this paper can be summarised as follows:

- An integrated QA-based RxM framework that combines ML-driven fault prediction with FMECA-KG reasoning to support automated root-cause analysis and maintenance recommendation.
- A rule-based mapping mechanism that links quantitative prediction results to qualitative maintenance knowledge, ensuring interpretability and traceability under limited data and small-corpus conditions.
- A practical validation through a case study, demonstrating the feasibility of the proposed approach on a linear actuator system under realistic data constraints.

Finally, although the present work deliberately avoids reliance on data hungry language models, it points towards a broader intelligent maintenance paradigm with significant extension potential. Future developments may include the incorporation of more advanced neural network (NN) architectures for data analysis or the selective integration of LLMs into rule interpretation and reasoning processes, thereby further enhancing the intelligence and adaptability of QA-based prescriptive maintenance systems.

1.5 | Paper Structure

The subsequent sections of this paper are organised as follows. Section 2 details the comprehensive methodological framework, outlining the synthesis of FMECA, KG and ML. Section 3 presents a critical analysis of the experimental results and evaluates the performance of the proposed system on the linear actuator case study. Section 4 synthesises the key findings of the research

and outlines future research directions for scaling QA-based prescriptive maintenance.

2 | Methodology

This section describes the architecture and configuration of each module within the proposed question-answering (QA) system. First, the construction of the failure modes, effects, and criticality analysis knowledge graph (FMECA-KG) from standardised FMECA documentation is presented. Next, the fault prediction module—referred to as the maintenance analytics tool—is introduced. Finally, the integration of the FMECA-KG and the fault prediction module for implementing the QA system is described.

2.1 | QA System Structure

Figure 3 illustrates the functional modules of the QA system and their interconnections. An engineered asset serves as the starting point of the system and provides two primary information sources: FMECA documentation and sensor measurement data.

FMECA documentation is generated using the IEC standard tabular template [13], based on domain experts' understanding of the asset's physical structure, functional characteristics and failure behaviour. This documentation is subsequently converted into a structured knowledge graph (FMECA-KG). In parallel, sensor data collected during asset operation are stored in a local database and accessed by the maintenance analytics tool, which autonomously learns from the data and produces fault prediction results.

When users interact with the QA system through the interface, their queries are analysed and mapped to predefined rules and templates. The system then retrieves relevant knowledge from the FMECA-KG and combines it with the fault prediction outputs to generate responses containing diagnostic and prescriptive maintenance information.

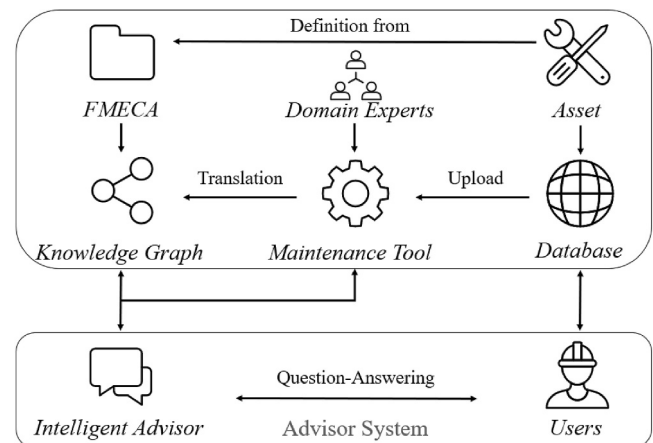


FIGURE 3 | Overall structure of the QA system.

2.2 | FMECA Knowledge Graph (KG)
Construction

2.2.1 | Asset Under Study and FMECA Development

A linear actuator, shown in Figure 4, was selected as the case study asset. A series of experiments was conducted under varying operating conditions and fault scenarios at Cranfield University [29]. The resulting datasets are publicly available and are briefly described in Section 2.3.1.

To establish the FMECA documentation for the linear actuator, key elements—including components, faults, failure modes, causes, effects and severity—occurrence—detection (SOD) ratings—were systematically defined. Following the standardised IEC tabular format [13], this information was recorded in text and structured tables. The resulting FMECA data serve as the foundational knowledge source for constructing the FMECA-KG, enabling automated reasoning and root-cause analysis through graph-based relationships.

2.2.2 | Entities and Relationships

In the FMECA documentation, entities correspond to either physical asset components (e.g., screws, balls and nuts) or specific fault types (e.g., spalling and backlash). Each entity belongs to a distinct class, such as Component, Fault, Failure, or Action, which is represented as a node label in the knowledge graph. Figure 5 illustrates the logical relationships between entities defined in the linear actuator FMECA.

In this study, the Neo4j platform was adopted to construct the FMECA-KG. Entity nodes were created and assigned labels according to their FMECA categories, while relationship properties were defined to represent the semantic links specified in the FMECA documentation. To enable efficient and reproducible KG construction, two CSV files were prepared following the entity–relationship logic illustrated in Figure 5. Table 1 presents a partial example of the FMECA knowledge encoded in these CSV files.

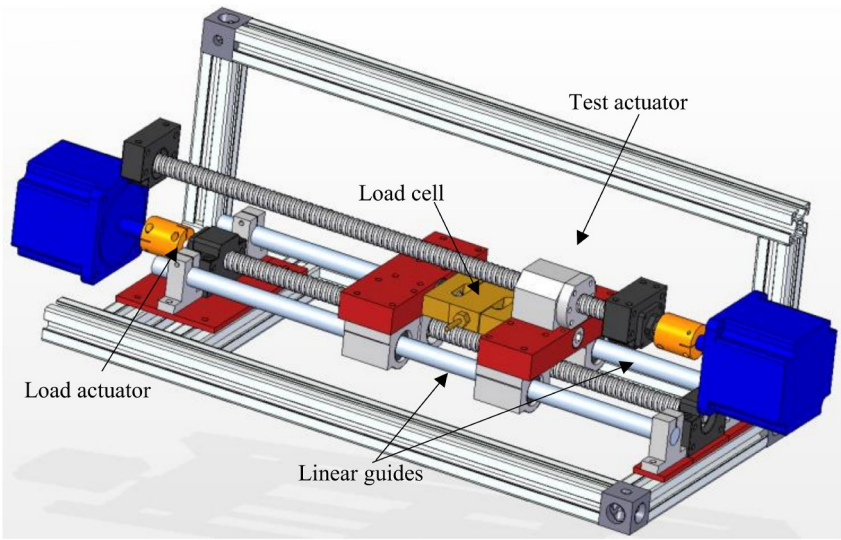


FIGURE 4 | Linear actuator used in the case study [29].

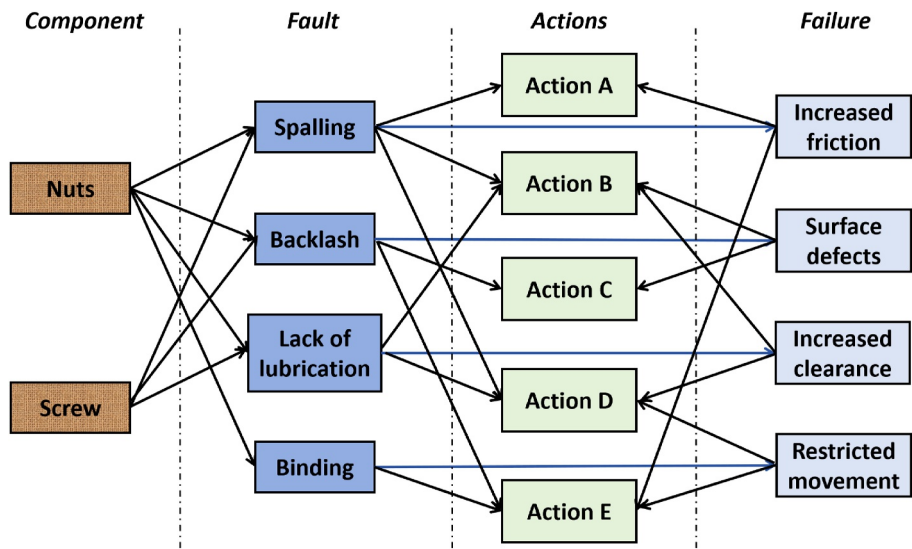


FIGURE 5 | Relationships between entities defined in the linear actuator FMECA.

During KG construction, entity types and relationship types were formally defined to reflect the structure of the FMECA methodology. In the FMECA-KG, entities such as failure modes, failure effects, mechanisms, and maintenance actions are represented as distinct node categories, interconnected through semantically meaningful relationships (e.g., `has_faults`, `has_effects` and `has_actions`). Table 2 summarises the entity and relationship types defined for the linear actuator FMECA-KG.

Based on these definitions, the relationships among different entity types in the FMECA-KG can be summarised as shown in Figure 6. The linear actuator is modelled as the root node, which connects to all its constituent components. For illustration, a single component is depicted in the figure to demonstrate how multiple sets of triples are associated with an individual element. Starting from a component, the KG enables traversal to identify potential faults, failure mechanisms, effects and corresponding maintenance actions.

As illustrated, the entities forming each triple are not isolated; instead, they collectively constitute a complex knowledge network. Within the FMECA-KG, fault analysis can be initiated from various entry points—such as a single fault symptom, a specific component or multiple simultaneous conditions—following the logic embedded in the FMECA structure. Figure 7 presents a partial view of the constructed FMECA-KG for the linear actuator implemented on the Neo4j platform.

TABLE 1 | FMECA knowledge defined in CSV files (partially shown).

Components	Faults	Actions	Relationships
Screws	Lack of lubrication	Lubrication check	has_faults
Balls	Spalling	Surface treatment	leads_to
Nuts	Backlash	Correct ball diameter	has_mechanism
...	...	...	...

TABLE 2 | Entities and relationships in the FMECA-KG.

Entities	Relationships
Components, Failure_of_Mechanism	has_actions_of_failures, has_failures
Description, Effects_of_Failures	has_actions_of_faults, has_faults
Equipment, actions	has_components, has_description_of_failures
Failures, faults	has_effects, has_description_of_faults
SOD_RPN, steps	leads_to, has_RPN, has_mechanism

2.3 | Maintenance Analytics

2.3.1 | Description of Measurement Signals

This study investigates machine learning (ML) based fault prediction using features extracted from measured operational signals to determine the health status or fault type in the linear actuator introduced in Section 2.2.1. The experimental dataset employed in this work originates from the test campaign reported in [29], in which four health status or fault type scenarios were considered: (i) normal condition, (ii) lack-of-lubrication, (iii) spalling, and (iv) backlash. For each scenario, datasets were collected under different loading conditions and motion trajectories to capture a broad range of operational behaviours.

As documented in [29], both the backlash and lack-of-lubrication fault types are characterised by two degradation levels, whereas the spalling fault type exhibits eight degradation levels. Since this study focuses on the early detection of fault conditions, only measurement data corresponding to the first degradation level of each fault type are analysed. Two representative motion trajectories, trapezoidal and sinusoidal, were imposed during the experiments. Both trajectories were evaluated under three distinct load conditions (20, 40, and –40 kg) in both healthy and faulty states. For each test condition, five complete motion cycles were performed and repeated 10 times, resulting in 50 complete motion cycles per condition.

In the trapezoidal motion trajectory, the actuator completed a 120 mm stroke at a constant velocity setpoint within 5 s, followed by a 3-s dwell period at each end of the stroke. In contrast, the sinusoidal trajectory featured smooth velocity transitions, completing the same 120 mm stroke in 6 s with a 2-s pause at each end. Each experiment, therefore, captured a full round—trip motion, extension and retraction, over a total duration of 16 s.

During each data acquisition, three signals were recorded to characterise the actuator's motion behaviour: (i) reference position, (ii) position error, and (iii) motor current. Figures 8 and 9 show representative examples of these three signals under trapezoidal and sinusoidal motion trajectories, respectively.

For each data acquisition, the three signals, namely reference position x , position error y , and current z , were sampled at a frequency of $f_s = 25$ Hz. Given the 16-s duration of each motion cycle (i.e., event), this resulted in $N = 400$ data points per signal per experiment. As all signals were digitised, they can be expressed as discrete time-series variables x_i , y_i , and z_i , where $i = 1, 2, \dots, N$ denotes the sample index. It is noted that the reference position signal x serves solely as a command input and remains identical across all experimental conditions, while variations in y and z reflect changes in system health and fault behaviour.

2.3.2 | Dataset Partitioning for RF Model Training–Testing

In this study, only the datasets acquired under the sinusoidal motion trajectory are considered. For these datasets, three loading conditions (20, 40, and –40 kg) and four operating states or fault types, namely (i) normal, (ii) backlash, (iii) spalling, and (iv) lack-of-lubrication, are investigated. Accordingly,

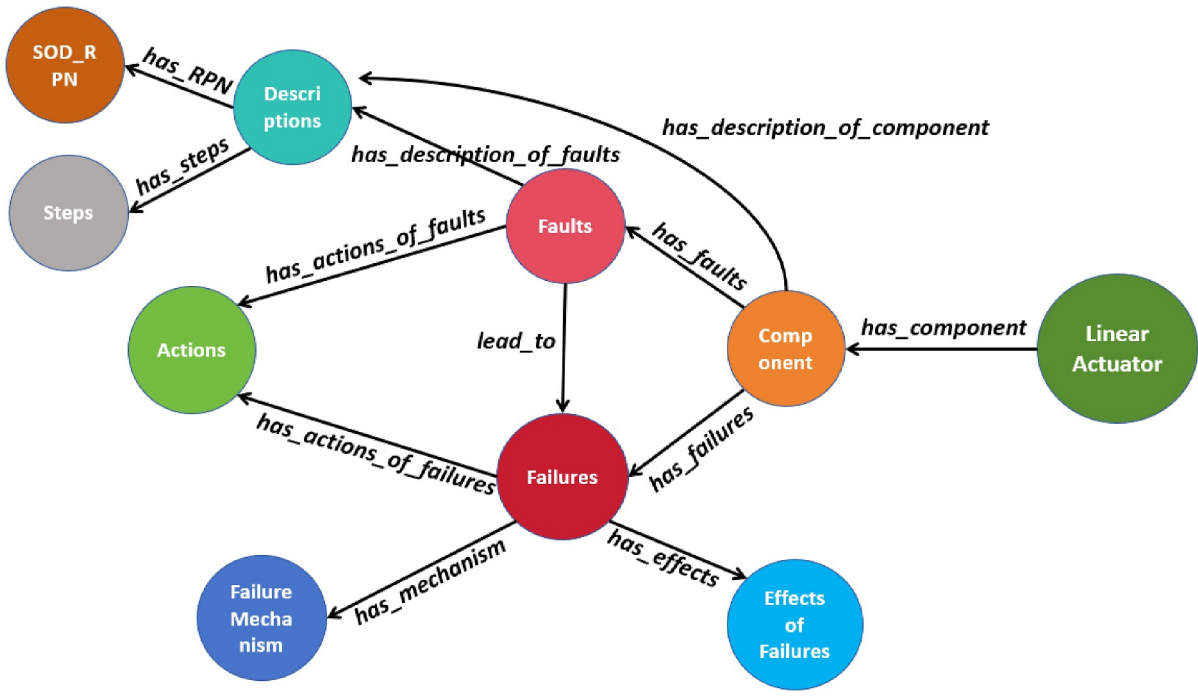


FIGURE 6 | Conceptual structure of the FMECA knowledge graph.

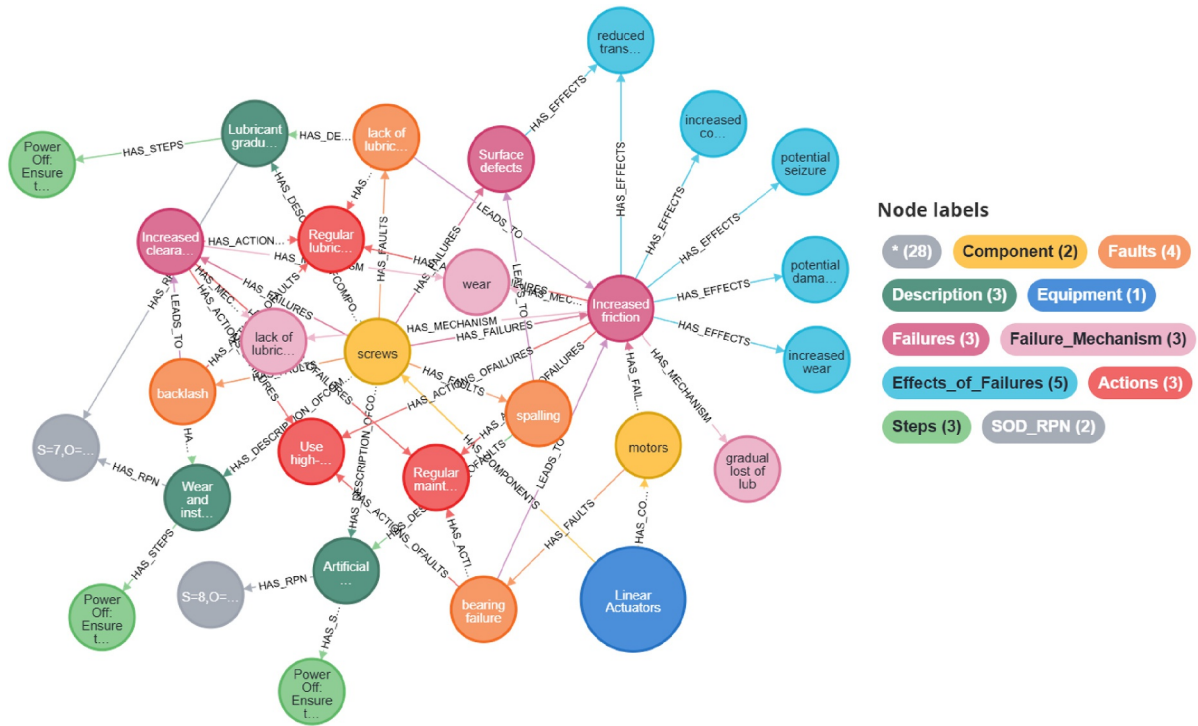


FIGURE 7 | FMECA-KG (partly shown).

a total of 12 test conditions are defined, as summarised in Table 3.

For each test condition, defined by a unique combination of fault type and applied load, 50 complete motion cycles (hereafter referred to as ‘events’) are available. The datasets are partitioned such that 80% of all the available motion cycles (i.e., 40 motion

cycles) are used for training, while the remaining 20% (i.e., 10 motion cycles) are reserved for testing. This data partitioning strategy is applied consistently across all test conditions, as illustrated in Table 3. Given the 12 defined test conditions, the overall dataset comprises $12 \times 50 = 600$ motion cycles, of which 480 motion cycles are used for model training and the remaining 120 motion cycles are used for model testing.

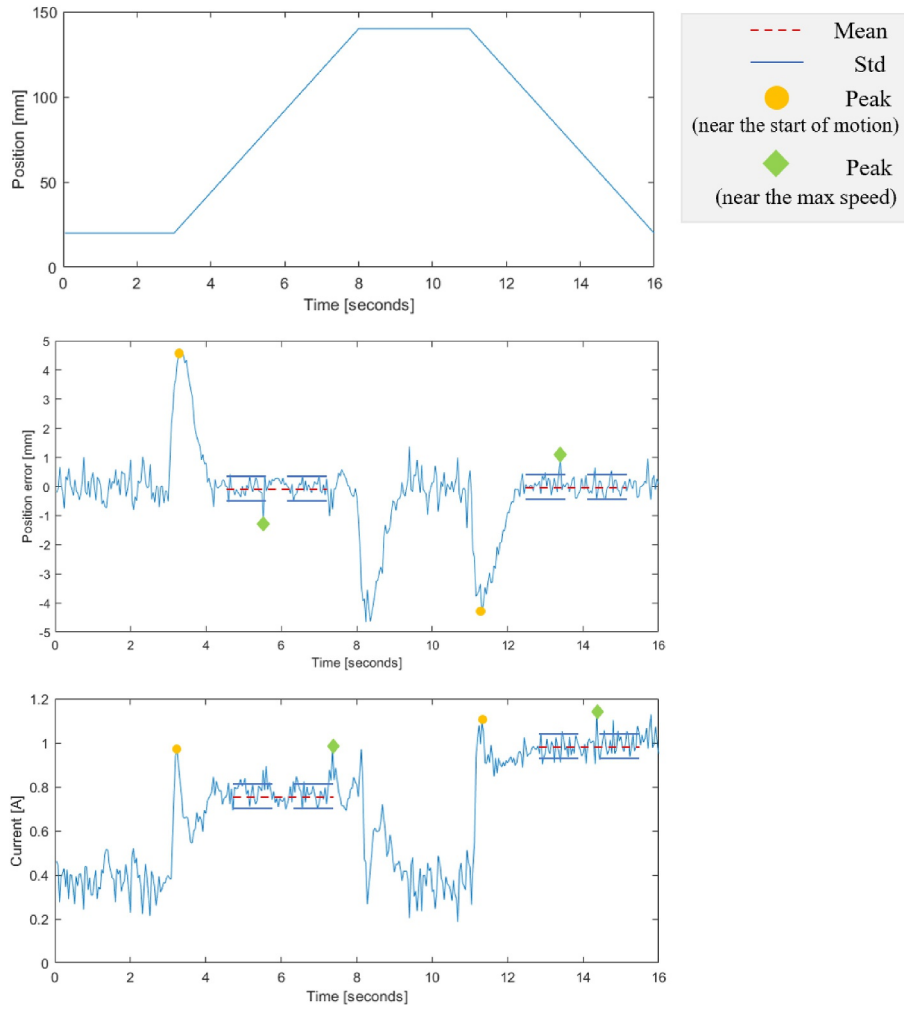


FIGURE 8 | Three representative signals under the trapezoidal motion trajectory.

2.3.3 | Signal Processing and Feature Engineering

In the fault analysis module of the proposed QA system, user-uploaded experimental data are processed to predict fault types through a four-stage workflow. First, representative signal features are extracted from the measured operational signals, as described in the following subsection. Second, Principal Component Analysis (PCA) is applied to the extracted feature set to reduce dimensionality and alleviate feature redundancy. Third, a Random Forest (RF) machine-learning model, briefly introduced in Section 2.3.4, is trained using the selected PCA components to perform fault classification. Finally, a hyper-parameter optimisation strategy, detailed in Section 2.3.5, is employed to further enhance the predictive performance of the RF model, with particular emphasis on evaluation metrics such as the macro $F1$ score.

Reliable fault prediction depends on both high-quality signal acquisition and effective feature extraction. Experimental results reveal noticeable differences in signal characteristics between trapezoidal and sinusoidal motion trajectories, as illustrated in Figures 8 and 9. For sinusoidal motion trajectories, the continuously varying velocity set-point prevents the signal

from reaching a stable steady-state value. In contrast, trapezoidal motion trajectories exhibit clear steady-state plateaus, which facilitate the extraction of localised signal features.

For the position error signal, the maximum peak value within a time window around motion start (including its temporal coordinate) and the overshoot value are selected as key features to characterise transient system behaviour. For the current signal, in addition to the overshoot, regions where the velocity variation around the maximum speed point is below 10% are analysed, as these intervals exhibit relatively stable current behaviour. From these regions, several features are extracted, including the mean steady-state value, the maximum peak value within a window around motion start, the maximum peak value within a window around maximum speed, and the standard deviation. However, due to the limited size of the training dataset, these localised signal features alone are insufficient for the RF model to reliably distinguish between similar fault conditions, particularly when steady-state current segments exhibit comparable feature values.

By contrast, sinusoidal motion trajectories require fewer localised features, resulting in lower computational complexity compared to trapezoidal trajectories. To improve the

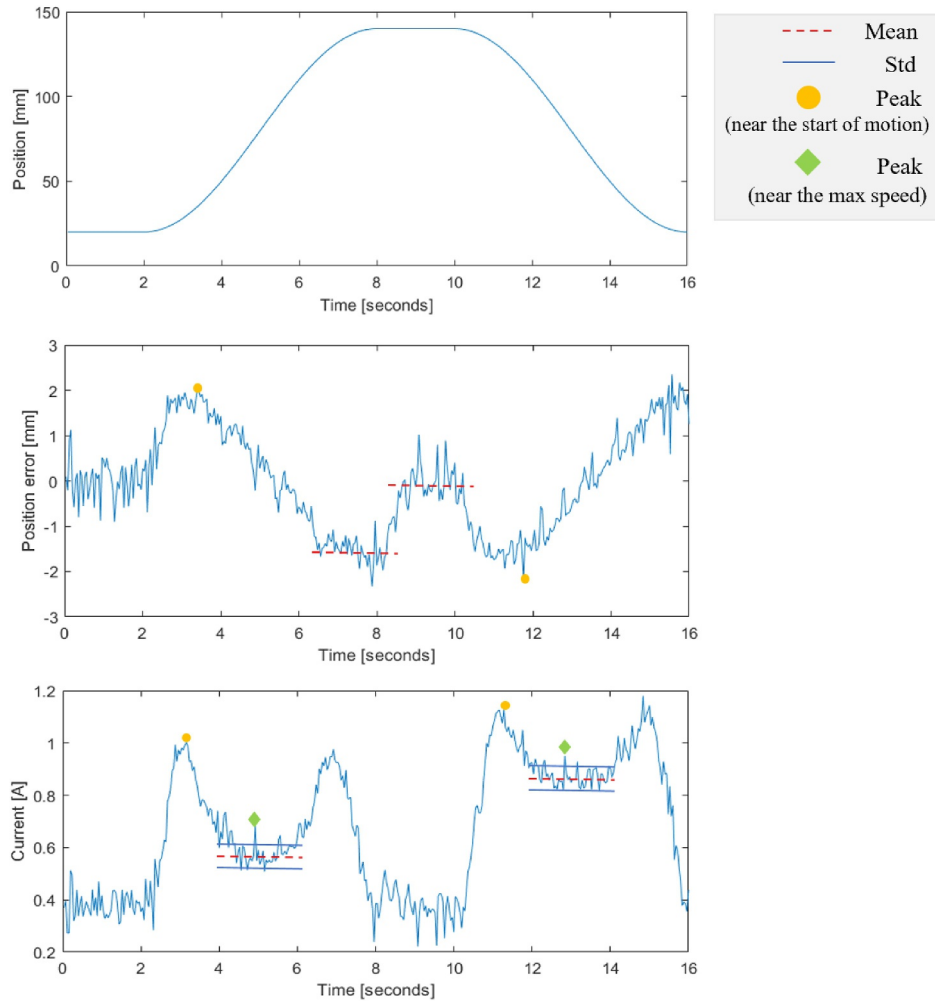


FIGURE 9 | Three representative signals under the sinusoidal motion trajectory.

TABLE 3 | The number of datasets for RF model training and testing.

No.	Label	Condition	Training set (#motion cycle/ event)	Testing set (#motion cycle/ event)
1	Normal20	Normal, load = 20 kg	40	10
2	Normal40	Normal, load = 40 kg	40	10
3	Normalneg40	Normal, load = -40 kg	40	10
4	Lackoflubrication20	Lack of lubrication, load = 20 kg	40	10
5	Lackoflubrication40	Lack of lubrication, load = 40 kg	40	10
6	Lackoflubricationneg40	Lack of lubrication, load = -40 kg	40	10
7	Spalling20	Spalling, load = 20 kg	40	10
8	Spalling40	Spalling, load = 40 kg	40	10
9	Spallingneg40	Spalling, load = -40 kg	40	10
10	Backlash20	Backlash, load = 20 kg	40	10
11	Backlash40	Backlash, load = 40 kg	40	10
12	Backlashneg40	Backlash, load = -40 kg	40	10

discriminative capability of the RF model for sinusoidal motion signals, additional global statistical features, namely Root Mean Square ($RMS_{y,z}$), Skewness ($SK_{y,z}$) and Crestor factors ($C_{y,z}$), are introduced. These features characterise the overall signal behaviour across the entire 16-s time window and complement the existing local features for enhanced fault separability. Table 4 summarises the features extracted from the two measured signals: the position error signal and the current signal.

In this study, feature extraction is restricted to the position error signal y_i and the current signal z_i . Each experimental test is conducted with multiple repetitions under varying operating conditions. To ensure computational consistency and balanced representation of the system's dynamic behaviour, signal features are extracted separately for the extension and retraction phases within each motion cycle.

As summarised in Table 4, a total of 26 features are extracted from the position error signal y_i and the current signal z_i for each motion cycle (i.e., event). For each test condition listed in Table 3, this corresponds to $40 \times 26 = 1,040$ feature samples used for model training, and $10 \times 26 = 260$ feature samples reserved for testing/prediction purposes. Considering all 12 test conditions, the complete training dataset therefore comprises $12 \times 1040 = 12,480$ feature samples, while the testing dataset consists of $12 \times 260 = 3,120$ feature samples.

2.3.3.1 | Overshoot Calculation. Overshoot is defined as the maximum magnitude by which the system response exceeds its initial steady-state value before convergence. For each motion cycle, overshoot is quantified as the difference between the peak response and the corresponding average steady-state response. The temporal locations of these features remain nearly consistent across all trials, as illustrated in Figures 10 and 11. For each motion cycle, the following intervals were identified:

1. Position error signal (y_i), extension phase: the peak response occurs between $2 - -4$ s ($i \in [50, 100]$), while the steady-state response is observed between $6 - -8$ s ($i \in [150, 200]$).
2. Position error signal (y_i), retraction phase: the peak response occurs between $10 - -12$ s ($i \in [250, 300]$), with the steady-state response between $8 - -10$ s ($i \in [200, 250]$).
3. Current signal (z_i): the first maximum peak value near the start of motion occurs within the interval of $2 - -4$ s ($i \in [50, 100]$), followed by a steady-state interval between $4 - -8$ s ($i \in [100, 150]$). The second maximum peak value near the start of motion is observed between $10 - -12$ s ($i \in [250, 300]$), with the corresponding steady-state interval occurring between $12 - -14$ s ($i \in [300, 350]$). For the maximum peak value near the maximum speed, the first occurrence is located within the

TABLE 4 | List of features calculated from the two signals for each motion cycle (i.e., event).

Signal type	Feature	Calculation scope
Position error (y_i)	Overshoot (OV_y)	Two features associated with retraction and extraction, respectively
	Max peak value near the start of motion (P_y)	Two features associated with retraction and extraction, respectively
	Max peak location near the start of motion (iP_y)	Two features associated with retraction and extraction, respectively
	RMS (RMS_y)	One feature associated with a complete motion cycle
	Skewness (SK_y)	One feature associated with a complete motion cycle
	Crest factor (C_y)	One feature associated with a complete motion cycle
Current (z_i)	Overshoot (OV_z)	Two features associated with retraction and extraction, respectively
	Max peak value near the start of motion (P_z)	Two features associated with retraction and extraction, respectively
	Max peak location near the start of motion (iP_z)	Two features associated with retraction and extraction, respectively
	Max peak value near the max speed (Ps_z)	Two features associated with retraction and extraction, respectively
	Max peak location near the max speed (iPs_z)	Two features associated with retraction and extraction, respectively
	Mean steady value (S_z)	Two features associated with retraction and extraction, respectively
	Standard deviation (σ_z)	Two features associated with retraction and extraction, respectively
	RMS (RMS_z)	One feature associated with a complete motion cycle
	Skewness (SK_z)	One feature associated with a complete motion cycle
	Crest factor (C_z)	One feature associated with a complete motion cycle

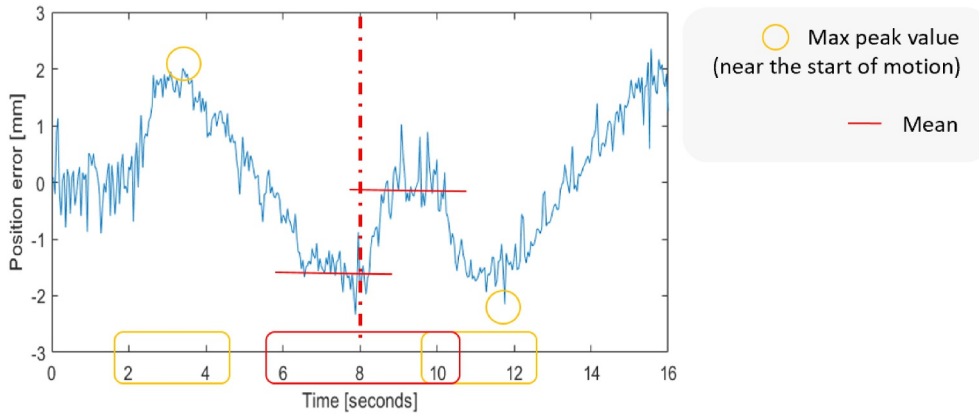


FIGURE 10 | Several features highlighted in a representative position error signal y_i .

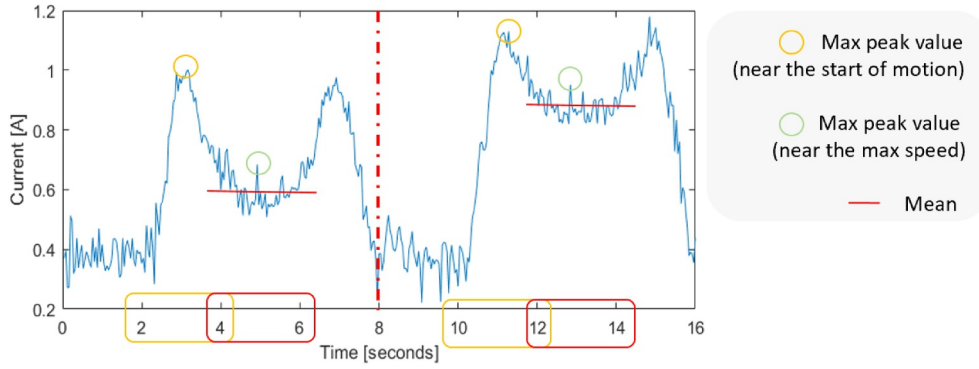


FIGURE 11 | Several features highlighted in a representative current signal z_i .

interval 4 – 6 s ($i \in [100, 150]$), while the second occurrence appears between 12 – 14 s ($i \in [300, 350]$).

Under these conditions, the overshoot of the position error signal, OV_y , and the overshoot of the current signal, OV_z , are approximated as:

$$OV_{y,z} = P_{y,z} - S_{y,z}, \quad (1)$$

where $P_{y,z}$ denotes the maximum peak value occurring near the start of motion within a single motion cycle of the signals y_i and z_i , respectively, and $S_{y,z}$ represents the corresponding average steady-state value computed over the subsequent 2 s interval. As illustrated in Figures 10 and 11, the maximum peak values near the start of motion and the steady-state values are highlighted in yellow and red, respectively. The peak and steady-state quantities are computed as:

$$P_y = \max_{i=j,\dots,k} y_i, \quad (2)$$

$$P_z = \max_{i=j,\dots,k} z_i, \quad (3)$$

$$S_y = \frac{1}{k-j+1} \sum_{i=j}^k y_i, \quad (4)$$

$$S_z = \frac{1}{k-j+1} \sum_{i=j}^k z_i, \quad (5)$$

where j and k are integer indices defining the lower and upper bounds of the sampling interval associated with each identified peak or steady-state region, as specified above.

For the maximum peak values near the maximum speed in the current signal z_i , the corresponding regions are highlighted in green, as shown in Figure 11. The peak value in this case is computed as:

$$Ps_z = \max_{i=j,\dots,k} z_i, \quad (6)$$

2.3.3.2 | Standard Deviation Calculation. The standard deviation is a statistical measure that quantifies the dispersion of a dataset relative to its mean. Since the time intervals during which the linear actuator reaches its maximum speed are known, the standard deviation is calculated only over the intervals of 4 – 6 and 12 – 14 s.

For the current signal z_i , these time intervals correspond to the integer sampling indices $i \in [100, 150]$ and $i \in [300, 350]$, respectively, as illustrated in Figure 12.

The standard deviation of the current signal, denoted by σ_z is calculated as:

$$\sigma_z = \sqrt{\frac{1}{k-j+1} \sum_{i=j}^k (z_i - \bar{z})^2}, \quad (7)$$

where $\bar{z}$ represents the mean value of the current signal z_i over the selected time interval $[j, k]$.

2.3.3.3 | Crest Factor and RMS Calculation. The crest factor is a time-domain feature used to characterise the peak

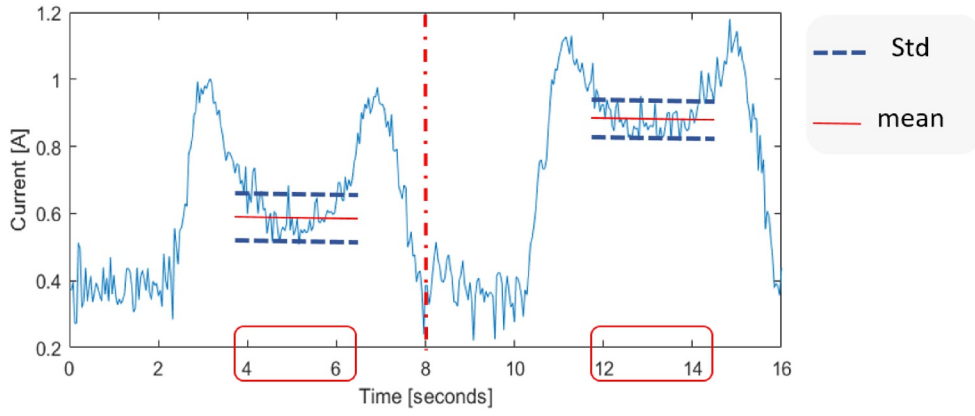


FIGURE 12 | Standard deviation feature highlighted in a representative current signal z_i .

magnitude of a signal relative to its energy content. It is defined as the ratio between the signal peak and its root-mean-square (RMS) value. The crest factor of either the y_i or the z_i signal, denoted by $C_{y,z}$ is given by:

$$C_{y,z} = \frac{P_{y,z}}{RMS_{y,z}}, \quad (8)$$

where $P_{y,z}$ denotes the peak value of the corresponding signal. The RMS values of the y_i and z_i signals are computed as:

$$RMS_y = \sqrt{\frac{1}{N} \sum_{i=1}^N y_i^2} \quad (9)$$

$$RMS_z = \sqrt{\frac{1}{N} \sum_{i=1}^N z_i^2} \quad (10)$$

where N is the total number of samples.

2.3.3.4 | Skewness Calculation. Skewness, denoted by $SK_{y,z}$, describes both the degree of symmetry and the direction of asymmetry of a data distribution. For the y_i signal, the skewness is calculated as:

$$SK_y = \frac{N}{(N-1)(N-2)} \sum_{i=1}^N \left(\frac{y_i - \bar{y}}{\sigma_y} \right)^3, \quad (11)$$

and for the z_i signal as:

$$SK_z = \frac{N}{(N-1)(N-2)} \sum_{i=1}^N \left(\frac{z_i - \bar{z}}{\sigma_z} \right)^3. \quad (12)$$

here $\bar{y}$ and $\bar{z}$ denote the mean values of the y_i and z_i signals, respectively, over the entire time interval ($i = 1, 2, \dots, N$), while σ_y and σ_z represent their corresponding standard deviations.

2.3.4 | Random Forest (RF)

Random Forest (RF) is an ensemble learning algorithm based on the bootstrap aggregating ('bagging') method, in which multiple decision trees are independently constructed using bootstrap samples drawn from the original dataset (input features), and predictions are obtained via majority voting [30]. Specifically, the algorithm repeatedly samples the training data with replacement, trains a classification or regression tree on

each sample, and predicts new instances by aggregating the outputs of the ensemble of trees [31]. An intuitive illustration of the bagging principle in RF is provided by Salman et al. [32]. In this study, we applied PCA to reduce dimensionality of features before the RF model, which is shown in Figure 13.

Decision trees constitute the fundamental building blocks of RF. A decision tree recursively partitions the input feature space into distinct regions based on splitting rules applied to the input features, and assigns a class label (or numerical value) at each leaf node. By aggregating a large number of diversified decision trees, RF can effectively capture the complex and nonlinear influence of different feature variables on the target output [33].

In the bootstrap sampling process, samples are drawn with replacement from the original dataset. When the number of draws equals the total number of samples M_s , the probability P_r that any given sample is selected at least once is given by:

$$P_r = 1 - \left(1 - \frac{1}{M_s} \right)^{M_s} \quad (13)$$

At each internal node of a decision tree, a random subset of features is selected from the full feature set, and the optimal feature from this subset is chosen to split the node. If the total number of available features is k , the number of candidate features d considered at each split is typically defined as:

$$d = \sqrt{k} \quad (14)$$

Each decision tree in the RF is trained independently using its corresponding bootstrap sample and feature subsets until a predefined stopping criterion is satisfied. In this study, the dataset is labelled according to the experimental descriptions introduced in Section 2.3.2. The trained RF model is subsequently embedded into the fault analysis module of the QA system, enabling automated prediction of fault types based on features extracted from measured signals.

Previous studies have demonstrated the effectiveness of RF in fault diagnosis applications. For example, Wu et al. [34] extracted multiple signal features and employed an RF-based feature selection strategy to identify an optimal subset, thereby improving classification performance. The performance of RF classifiers is commonly evaluated using Precision, Recall,

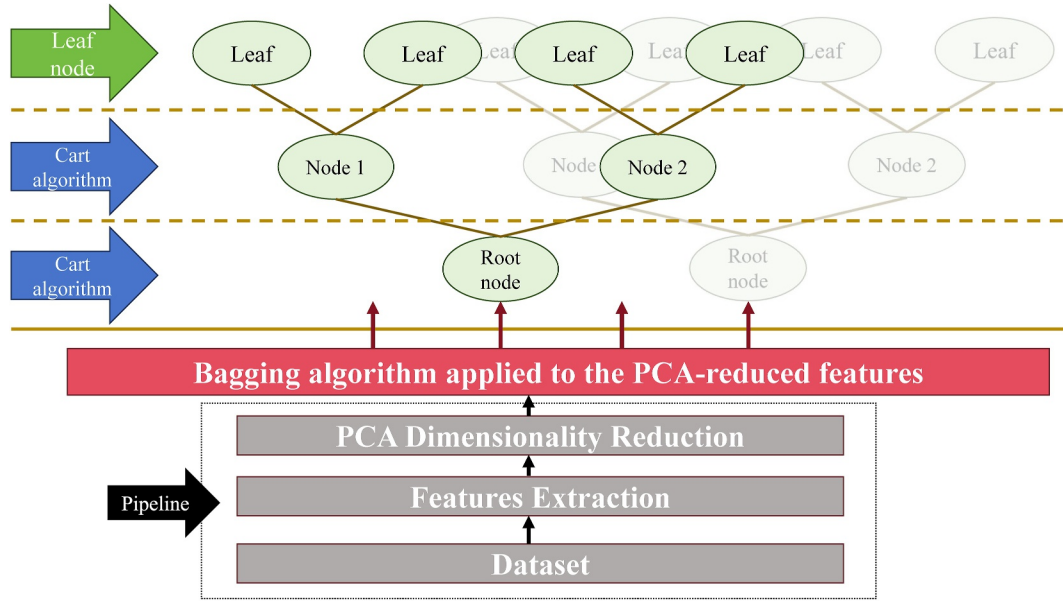


FIGURE 13 | Schematic diagram of the RF algorithm with a pre-processing PCA. *Source:* Adopted from [32].

and the $F1$ score. Precision represents the proportion of samples predicted as positive that are truly positive, while Recall measures the proportion of actual positive samples that are correctly identified. These metrics are defined as:

$$\text{Precision} = \frac{TP}{TP + FP} \quad (15)$$

$$\text{Recall} = \frac{TP}{TP + FN} \quad (16)$$

where TP denotes the number of true positives, FP the number of false positives, and FN the number of false negatives.

The $F1$ score provides a balanced measure of classification performance by computing the harmonic mean of Precision and Recall:

$$F1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (17)$$

Since RF performs multi-class classification in this study, the macro $F1$ score, defined as the arithmetic mean of the $F1$ scores across all classes, is adopted as the primary evaluation metric for hyperparameter optimisation, as discussed in the subsequent section.

2.3.5 | Hyperparameter Optimisation

The performance of the Random Forest (RF) model in distinguishing among different fault types depends on both the structural configuration of the individual decision trees and the characteristics of the extracted signal features. These design choices, together with the selected feature dimensionality, constitute key hyperparameters of the RF model. Appropriate tuning of these hyperparameters is therefore essential to achieve optimal fault classification performance.

Bayesian optimisation (BO) [35] is an efficient and widely adopted strategy for hyperparameter tuning [36]. The RF algorithm includes several user-defined parameters that influence classification performance. In this study, the random seed was fixed to 42 to ensure reproducibility. Among the available hyperparameters, two were identified as having a substantial impact on the RF model's performance: (i) the number of trees (i.e., estimators) in the forest and (ii) the number of Principal Component Analysis (PCA) components retained for dimensionality reduction. Consequently, these two hyperparameters were selected for optimisation.

The objective of the BO process was defined as the maximisation of the macro-averaged $F1$ score, which provides a balanced evaluation of classification performance across all fault classes. Within the BO framework, the objective function was formulated as a function of the RF hyperparameter configuration, allowing systematic exploration of the hyperparameter search space.

To model the relationship between hyperparameter combinations and the resulting macro $F1$ score, a Gaussian Process (GP) surrogate model was employed. Further details on GP modelling for hyperparameter optimisation can be found in [37, 38]. A dynamic acquisition strategy was then adopted, which probabilistically selects among multiple acquisition functions, including the Lower Confidence Bound (LCB), Expected Improvement (EI), and Probability of Improvement (PoI). Through this iterative process, the BO algorithm identifies the optimal hyperparameter configuration θ^* by maximising the objective function $f(\theta)$:

$$\theta^* = \underset{\theta \in \mathbb{R}^p}{\operatorname{argmax}} f(\theta) \quad (18)$$

where θ denotes the vector of hyperparameters to be optimised and $\mathbb{R}^p$ represents the p -dimensional real-valued hyperparameter space, with $p = 2$ in this study. For further details on

the acquisition functions employed, readers are referred to the relevant literature on LCB [39], EI [40], and PoI [39, 41].

2.4 | The Advisor (QA) System

2.4.1 | Advisor Structure

The QA system comprises two parallel data paths, as illustrated in Figure 14. The first path processes user statements or queries, while the second handles sensor data.

In the first data path, the QA system processes user queries on the back end using Natural Language Processing (NLP) techniques. Both Knowledge Graphs (KGs) and NLP have been widely adopted to efficiently manage and interpret language resources [42]. In this study, the primary objective of the NLP module is to transform user queries related to the health status and fault root causes of a linear actuator into Cypher queries, which can then be executed over the FMECA-KG.

In the second data path, the QA system automatically identifies potential fault types in the linear actuator based on sensor data. Once potential faults are detected, the system follows the same processing workflow as in the first data path. In addition, this automated analysis generates and returns visual plots to illustrate the corresponding data analysis results.

For the development of the NLP module, this study adopts open-source code from an existing QA system built on a medical KG (GitHub project [43]). Using a rule- and dictionary-based approach, the NLP processing pipeline consists of the following three main stages:

1. Segment the words in the user's query using a custom dictionary and word segmentation techniques.
2. Extract keywords, categorise them and assign keyword types (e.g., component, fault or failure) using rule-based methods.

3. Detect interrogative terms, invoke the corresponding Cypher query templates through rule-based matching, and return the generated query to the system.

Once the QA system receives the constructed Cypher query, it retrieves the relevant entities, attributes and relationships from the FMECA-KG. These results are then incorporated into a predefined answer template to generate the final response.

2.4.2 | Named Entities Recognition

Named entity recognition (NER) is employed to identify and categorise entities with specific semantic meanings within the text. In this work, NER is aligned with the entity types defined in the FMECA documentation. Based on frequently asked fault-related questions, a dictionary is constructed to enable automatic entity recognition by the system, thereby facilitating subsequent query classification. The entity categories included in this dictionary are summarised in Table 5.

To identify the predefined keywords, the Aho–Corasick (AC) algorithm is employed. The AC algorithm is a widely used method for efficiently matching multiple string patterns simultaneously [44]. It locates all occurrences of multiple keywords by scanning the input text only once. By applying the AC algorithm together with the dictionary listed in Table 5, the QA

TABLE 5 | Types in the dictionary.

From FMECA	From question
Action class	Problem: 'damage', 'alarm'...
Component class	Component: 'element', 'piece'...
Effect class	Cause: 'why', 'reason'...
Failure class	Prevent: 'avoid', 'stop'...
Fault class	Repair: 'fix', 'rebuild'...
Mechanism class	Definition: 'Description'...

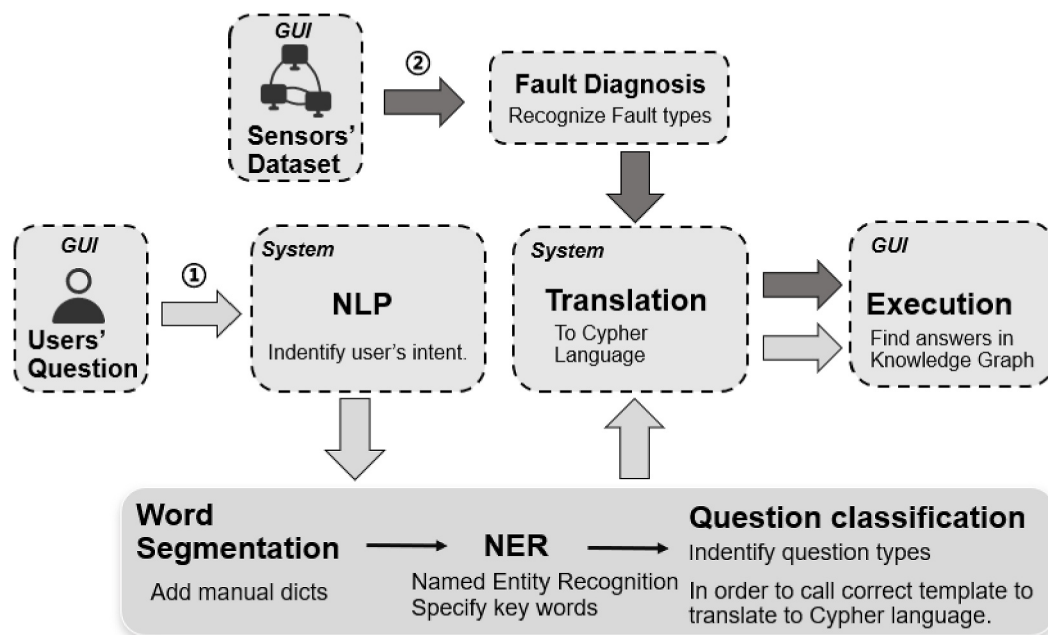


FIGURE 14 | Structure of the QA system.

system can associate recognised fault-related terms with their corresponding entities in the FMECA-KG.

2.5 | Graph Searching

After the QA system identifies the question type, it generates the corresponding Cypher query and executes it on the FMECA-KG. Neo4j then returns the relevant entities, attributes and relationships, which are subsequently populated into a predefined answer template. The overall search process is illustrated in Figure 15.

Similarly, once the fault data analysis results are obtained, the system converts the identified fault labels into textual form and inputs them into the QA system. The same NLP procedure is then executed to extract the relevant entities, attributes and relationships, as illustrated in Figure 16.

3 | Results and Discussion

This section first examines the findings derived from the FMECA documentation and describes the construction and

organisation of the proposed FMECA-KG. Subsequently, the effectiveness of the random forest (RF) method is evaluated, and the corresponding fault-type prediction results are presented. The prediction outcomes are analysed using both quantitative metrics and visual representations. Finally, an overview of the question-answering (QA) system interface is provided.

3.1 | The FMECA Knowledge Graph

3.1.1 | The FMECA Documentation

The FMECA documentation defines 10 entity classes, as summarised in Table 6. According to the entity and relationship definitions presented in Table 2, the *Component* entity consists of 12 distinct part names. The *Fault* entity includes 23 fault types, whereas the *Failure* entity contains 20 failure types. In total, 35 failure mechanisms are categorised under the *Failure Mechanism* entity, and 32 effects are captured within the *Effects of Failure* entity.

Furthermore, the *Action* entity comprises 34 corrective or preventive measures. The *Risk Priority Number* (RPN) entity

Question	Key words & Types	Cypher Language
<i>what is the cause of spalling</i>	Key words: 'what' 'what is' 'cause' 'spalling' Types: 'definition_fault' 'cause_fault'	<pre> 'definition_fault': MATCH (m:Faults)-[r:HAS_DESCRIPTION_OFFAULTS]->(n:Description) where m.attribute_name = 'spalling' return m.attribute_name, n.attribute_name 'cause_fault': MATCH (m:Faults)-[r:HAS_FAULTS]-(n:Component) where m.attribute_name = 'spalling' return m.attribute_name, n.attribute_name, r.name + MATCH (m:Faults)-[r:HAS_DESCRIPTION_OFFAULTS]->(n:Description) where m.attribute_name = 'spalling' return n.attribute_name, r.name </pre>
Answers	Graph returns	
<pre> 'definition_fault': The fault spalling has following descriptions: Surface defects cause irregular movement and increased wear;... 'cause_fault': The fault spalling may come from component: nuts;screws;balls;bearing. The detail of these faults are respectively: Surface defects cause irregular movement and increased wear;... </pre>		

FIGURE 15 | Searching processes in KG.

incorporates severity, occurrence and detection (SOD) values for 11 distinct combinations. The *Step* entity contains 38 sets of detailed inspection or maintenance procedures. Table 7 presents representative examples extracted from the FMECA documentation.

3.1.2 | Knowledge Graph Structure

The FMECA-KG is constructed based on the FMECA documentation described above. Figure 17 illustrates the overall structure and semantic relationships within the FMECA-KG.

The resulting FMECA-KG contains 10 node label types and 12 relationship types. The distribution of entity nodes is summarised in Table 6, whereas the number of relationships for each relation type is reported in Table 8. These statistics highlight the richness and completeness of the constructed knowledge graph for failure mode and maintenance analysis.

3.2 | Maintenance Datasets Analytics Results

3.2.1 | Optimised Hyperparameters for RF Model

Following the hyperparameter optimisation procedure described in Section 2.3.5, a predefined search space was specified for each

Label returns	Key words & Types	Cypher Language
'spalling' or 'backlash' or 'lack of lubrication' or 'normal'	Key words (example): 'backlash' Types: 'exist_fault'	'exist_fault': MATCH (m:Faults)-[r:HAS_FAULTS]-(n:Component) where m.attribute_name = 'backlash' return m.attribute_name, n.attribute_name, r.name + MATCH (m:Faults)-[r:HAS_DESCRIPTION_OF_FAULTS]- >(n:Description) where m.attribute_name = 'backlash' return n.attribute_name, r.name
Answers	Graph returns	
'definition_fault': All prediction here: ['Backlash20',... 'Backlash20'] I get label: backlash The fault backlash may come from component: bearings;nuts;balls;screws. The detail of these faults are respectively: Smaller balls cause increased play and transmission error;..		

FIGURE 16 | Searching processes for fault diagnosis results.

TABLE 6 | Number of entities in the FMECA-KG.

Node label	Count
Actions	34
Component	12
Description	38
Effects_of_Failures	32
Equipment	1
Failure_Mechanism	35
Failures	20
Faults	23
SOD_RPN	11
Steps	38

TABLE 7 | The FMECA documentation (partly shown).

Components	Faults	Failures	Actions, ...
Screw	Lack of lubrication	Increased friction	Apply lubrication, ...
	Spalling	Surface defects	Surface treatments, ...
Balls	Backlash	Increased clearance	Correct ball diameter, ...
	Contamination	Increased wear	Inspect regularly, ...
Nuts	Over tightened	Restricted movement	Seal adjustment, ...
	Backlash	Increased clearance	Check per installation, ...

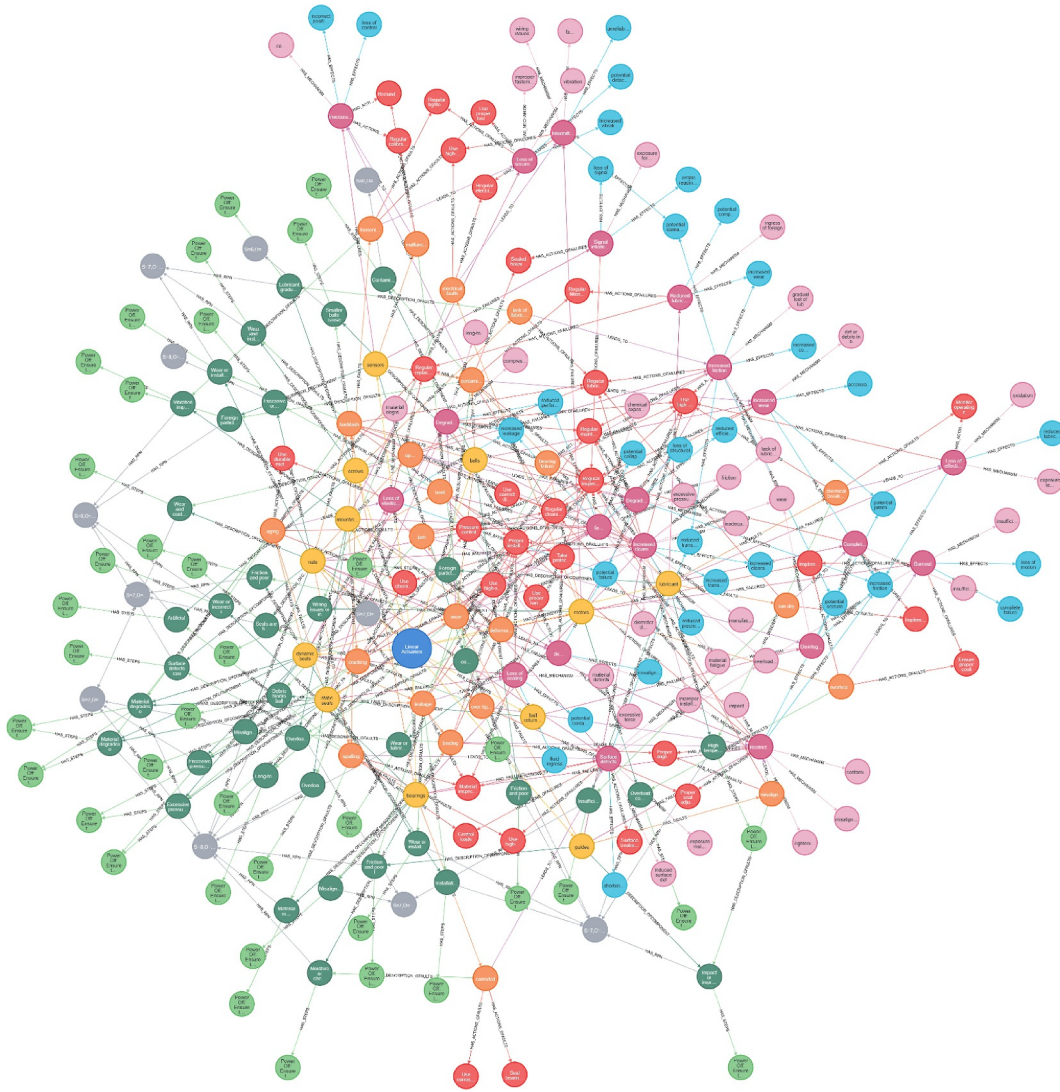


FIGURE 17 | Overall structure of the FMECA-KG.

TABLE 8 | Numbers of relationships.

Relationships	Count
HAS_ACTIONS_OF_FAILURES	57
HAS_ACTIONS_OF_FAULTS	57
HAS_COMPONENTS	12
HAS_DESCRIPTION_OF_FAULTS	38
HAS_DESCRIPTION_OF_COMPONENTS	42
HAS_EFFECTS	48
HAS_FAILURES	36
HAS_FAULTS	38
HAS_MECHANISM	47
HAS_RPN	38
LEADS_TO	26
HAS_STEPS	42

model hyperparameter. Under the condition where the random seed was fixed at 42, the number of trees (i.e., estimators) was varied within the range of 50–250, whereas the number of

principal components was explored within the range of 4–8. Applying this optimisation strategy to the training feature set yielded the optimal combination of hyperparameters, with the corresponding performance results summarised in Table 9.

3.2.2 | Prediction Results

Following the data analytics methodology described in Section 2.3, the RF model embedded in the QA system was trained using features extracted from signals acquired over 480 motion cycles (i.e., events), as summarised in Table 3. The trained RF model was then deployed within the QA system and evaluated using a test dataset comprising 120 motion cycles (i.e., events). This evaluation assessed overall prediction performance under varying load conditions across 12 fault labels. The resulting confusion matrix is shown in Figure 18.

As shown in the figure, the overall precision of the RF model reached 85.00%. A comparatively lower prediction performance is observed for the spalling fault, whereas the model demonstrates strong performance for the remaining three fault types. Notably, all lack-of-lubrication samples were correctly classified.

For samples corresponding to normal operating conditions, one sample under each load condition was misclassified as spalling with the same load label. In the case of backlash faults, three samples under the -40 kg load condition were misclassified as spalling with the corresponding load label. For spalling fault samples, multiple misclassifications occurred, primarily as normal or backlash conditions. To enable further comparative analysis, the evaluation metrics for the four fault conditions are summarised in Figure 19.

Among the 120 motion cycles (i.e., test samples), the RF model achieved the highest recall for the backlash and lack-of-lubrication fault types, followed by the normal operating condition. In contrast, the spalling fault exhibited the poorest recall performance.

A plausible explanation is that the initial degradation level associated with the spalling fault is insufficiently pronounced, resulting in extracted feature characteristics that are relatively similar to those of normal conditions and backlash faults. This overlap in feature space can negatively impact the model's discriminative capability for early-stage spalling. A more detailed investigation will be carried out in the subsequent

discussion, in conjunction with the signal curves generated by the QA system.

3.2.3 | Normal Condition

Figure 20 illustrates representative signals obtained under normal operating conditions. The blue curves correspond to the measured position error and motor current signals. For comparison, the expected reference signals under normal conditions and the same load are plotted as green curves within the same time interval.

Four characteristic features are highlighted using different colours: overshoot, maximum peak value, mean value and standard deviation (σ). These features are annotated separately for the extension and retraction phases, denoted by 'ex' and 're', respectively. In the title bar, the prefix 'I-' indicates the predicted fault type for the individual data instance, whereas 'A-' represents the aggregated prediction result for the dataset group to which the instance belongs.

For the normal condition, the predicted signals closely match the reference curves, demonstrating strong agreement between the measured data and the model expectations. This alignment confirms that the RF model correctly identifies normal behaviour, as anticipated.

TABLE 9 | Optimised hyperparameters and evaluation values.

Optimised hyperparameters			
Number of trees	51		
Number of PCA components	7		
Evaluation values of fault conditions			
Fault type	Precision	Recall	F1-score
Normal condition	0.8155	0.8000	0.8077
Backlash	0.8515	0.8190	0.8350
Spalling	0.7522	0.8095	0.7798
Lack of lubrication	0.9806	0.9619	0.9712
Overall evaluation values			
Macro average	0.8500	0.8476	0.8484

3.2.4 | Lack of Lubrication

Figure 21a presents the position error signal under a lack-of-lubrication condition, where a noticeably larger absolute peak value is observed compared with the normal operating case. In addition, the corresponding current signal shown in Figure 21b exhibits higher peak values, increased mean values and a larger standard deviation (Std), indicating greater signal fluctuation.

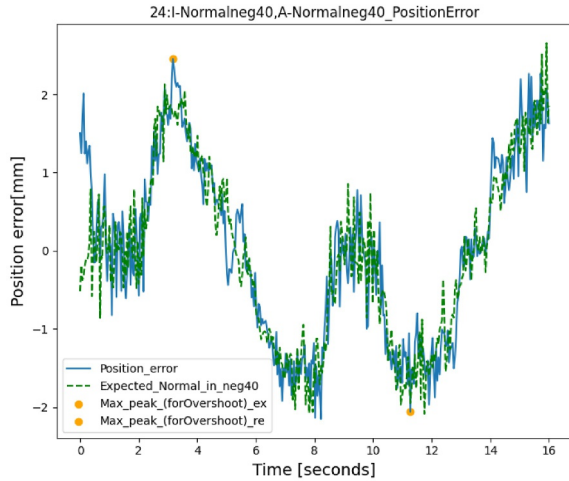
These pronounced deviations from normal behaviour enhance the separability of this fault type in the feature space, thereby facilitating more accurate classification by the prediction model. Consequently, the RF model achieves improved prediction performance for this fault category, as reflected in the results shown in Figure 18.

Prediction Label \ Real Label (set)	Normal 20	Normal 40	Normal -40	Backlash 20	Backlash 40	Backlash -40	Spalling 20	Spalling 40	Spalling -40	Lack lubrication 20	Lack lubrication 40	Lack lubrication -40
Normal 20	8	0	0	0	0	0	0	0	0	0	0	0
Normal 40	0	9	0	0	0	0	0	3	0	0	0	0
Normal -40	0	0	8	0	0	0	0	0	1	0	0	0
Backlash 20	1	0	0	10	0	0	1	0	0	0	0	0
Backlash 40	0	0	0	0	10	0	0	2	0	0	0	0
Backlash -40	0	0	1	0	0	7	0	0	2	0	0	0
Spalling 20	1	0	0	0	0	0	8	0	0	0	0	0
Spalling 40	0	1	0	0	0	0	1	5	0	0	0	0
Spalling -40	0	0	1	0	0	3	0	0	7	0	0	0
Lack lubrication 20	0	0	0	0	0	0	0	0	0	10	0	0
Lack lubrication 40	0	0	0	0	0	0	0	0	0	0	10	0
Lack lubrication -40	0	0	0	0	0	0	0	0	0	0	0	10
Total Prediction Label	10	10	10	10	10	10	10	10	10	10	10	10
Total Precision	8+9+8+10+10+7+8+5+7+10+10+10 / 120 = 85.00%											

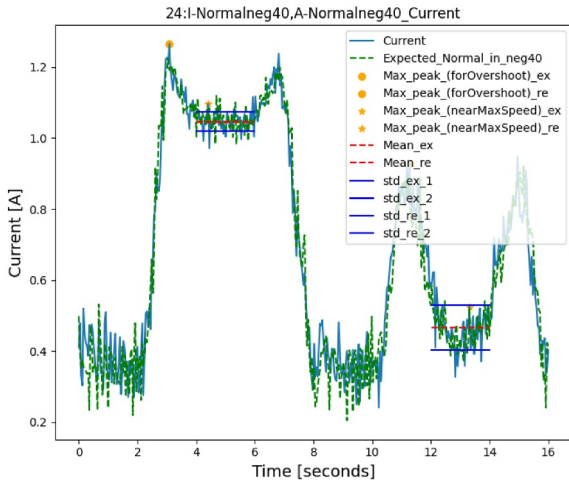
FIGURE 18 | Prediction results of 12 fault labels.

Total Motion Cycles for Prediction = 120				
Prediction Label \ Real Label (set)	Normal	Backlash	Spalling	Lack of lubrication
Normal	25	0	4	0
Backlash	2	27	5	0
Spalling	3	3	21	0
Lack of lubrication	0	0	0	30
Precision	86.21%	79.41%	77.78%	100%
Recall	83.33%	90.00%	70.00%	100%
F1 Score	84.75%	84.38%	73.68%	100%

FIGURE 19 | Prediction results of four fault conditions.



(a)

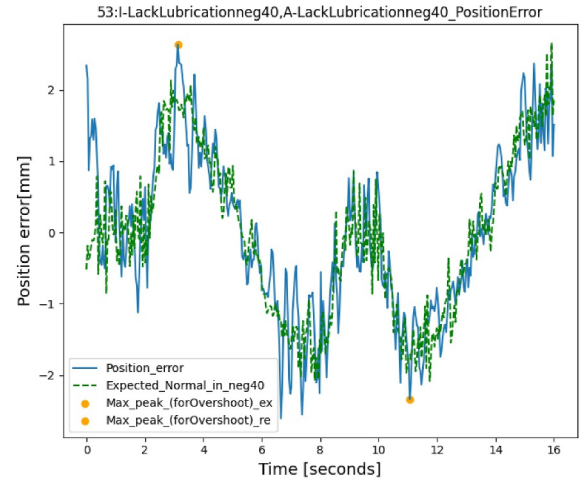


(b)

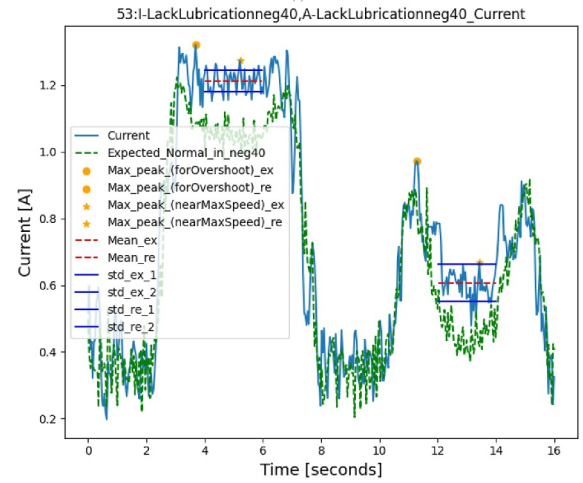
FIGURE 20 | Representative measured signals in a normal state, (a) position error signal and (b) current signal.

3.2.5 | Backlash

Figure 22 demonstrates that during the extension phase, the linear actuator signals under a backlash fault exhibit behaviour that is largely comparable to that of the normal operating condition. In contrast, pronounced deviations become evident during the retraction phase. As shown in Figure 22a, the position error signal associated with the backlash fault displays a



(a)

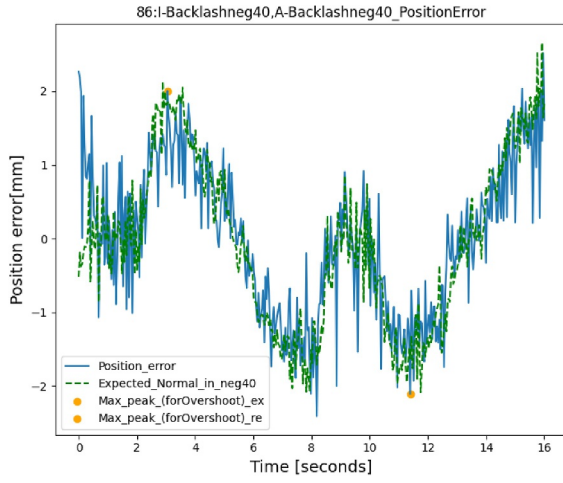


(b)

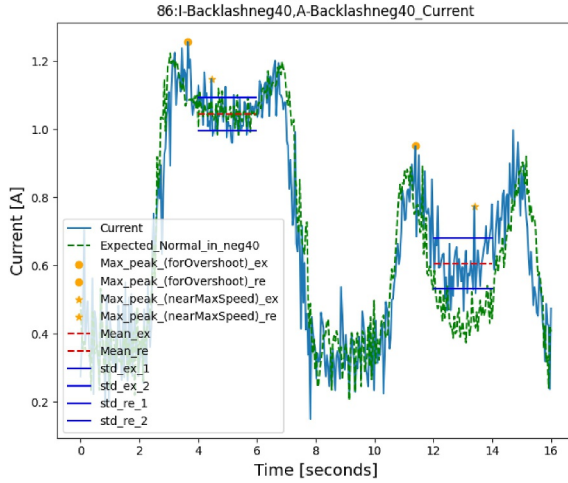
FIGURE 21 | Representative measured signals under lack-of-lubrication, (a) position error signal and (b) current signal.

significantly higher absolute overshoot than that observed under normal conditions.

Similarly, the current signal presented in Figure 22b shows increased values across all extracted features when compared with the normal case, with these differences being particularly pronounced during the retraction phase. Compared with other fault types, the backlash condition therefore exhibits more



(a)



(b)

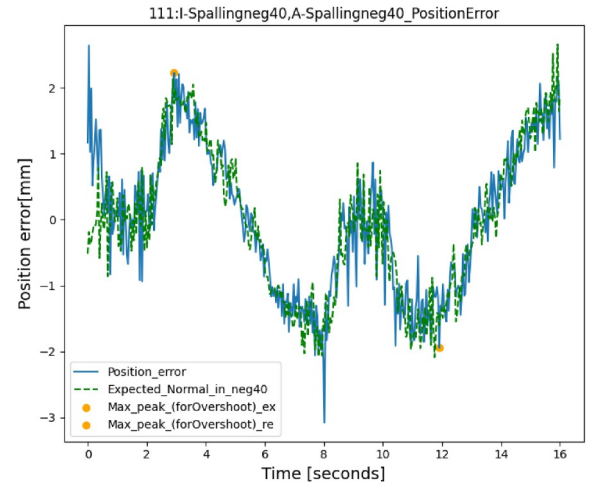
FIGURE 22 | Representative measured signals under backlash condition, (a) position error signal and (b) current signal.

distinguishable signal characteristics relative to normal operation, specifically during retraction, enabling the prediction model to effectively discriminate between normal and backlash fault states.

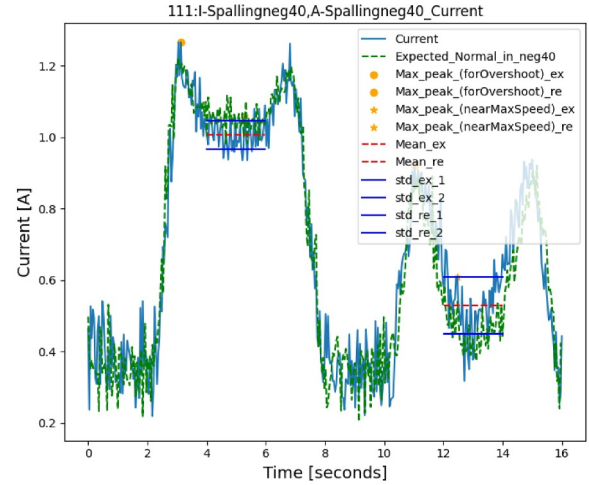
3.2.6 | Spalling

Figure 23 shows that the position error signals observed during both the extension and retraction phases under the spalling condition closely resemble those recorded under normal operating conditions. Moreover, the current signals during the first extension movement exhibit similarly high conformity with the normal case. In contrast, during the second retraction movement, the current signal demonstrates noticeable increases in standard deviation, overshoot and peak value.

These observations indicate that, under the first degradation level of the linear actuator, the time-domain characteristics associated with spalling faults are generally very similar to those of normal operation. Although higher degradation levels could make the fault characteristics more pronounced and enhance feature distinguishability, they may also introduce additional uncertainty and variability in the signal patterns, thereby



(a)



(b)

FIGURE 23 | Representative measured signals under spalling fault condition, (a) position error signal and (b) current signal.

increasing the overall complexity of the classification task. Consequently, despite its overall effectiveness, the RF model employed in this study exhibits a performance bottleneck when distinguishing the spalling fault class.

3.3 | Interaction With the Advisor System

To facilitate the practical use of the integrated fault analysis functionality, a front-end interface or graphical user interface (GUI) was developed for the advisor system, as illustrated in Figure 24. The interface supports two distinct query modes for user interaction.

In the first mode, users submit queries in natural language. Upon receiving a query, the system retrieves relevant information from the FMECA-KG and, where applicable, performs fault diagnosis based on the associated sensor data. Representative interactions using this mode are shown in Figure 25.

In the second mode, the users upload measured sensor data directly. The pre-trained random forest (RF) model described earlier processes the uploaded sensor data to predict the corresponding fault type and generates the associated fault labels.

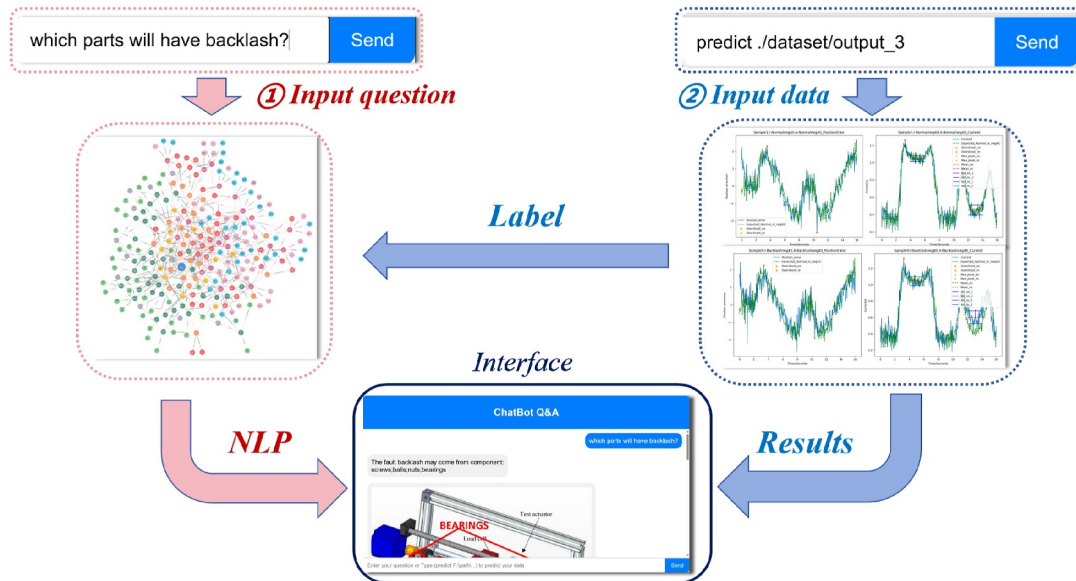


FIGURE 24 | Input interface of the advisor system.

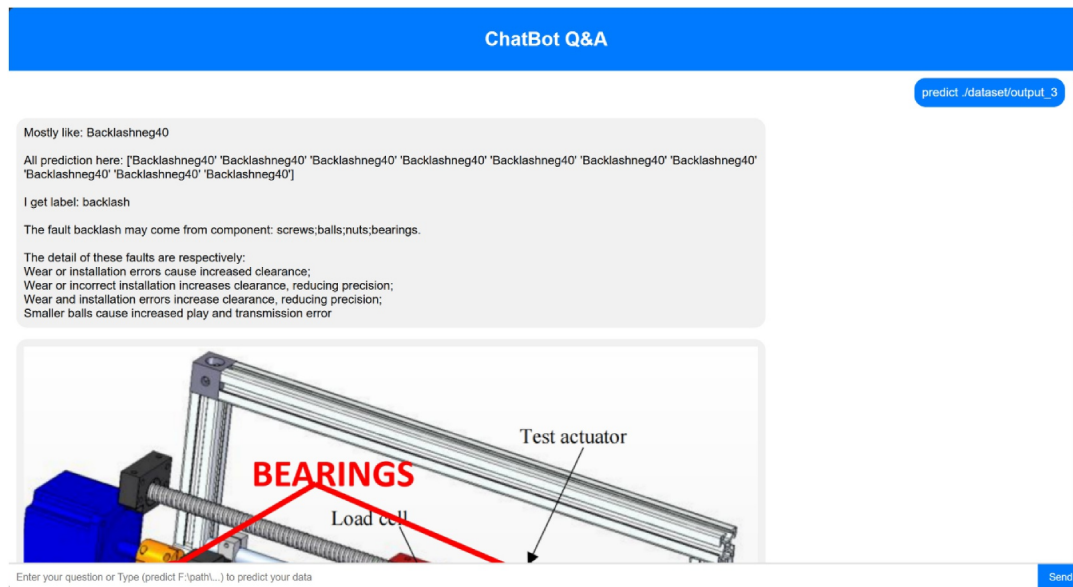


FIGURE 25 | Advisor interaction (part 1).

The system then follows the same processing pipeline as in the first mode: Querying the FMECA-KG, compiling diagnostic and explanatory information and presenting the results through the front-end interface. Representative outputs are shown in Figures 25 and 26.

Within the interface, the advisor system first displays the fault prediction results for the submitted data. It then reports the predicted fault labels and retrieves the corresponding diagnostic knowledge from the FMECA-KG. To enhance interpretability, a component position diagram of the linear actuator is generated to provide spatial context. Finally, all diagnostic, explanatory and visual information related to the analysed dataset is consolidated and presented to the user for review.

4 | Conclusion and Outlook

4.1 | Conclusions

This study has investigated the design and implementation of an artificial intelligence (AI) or machine learning (ML)-enabled advisory system to support prescriptive maintenance in engineered systems. Framed within the context of Question–Answering (QA) systems, the proposed approach combines data-driven fault diagnosis with knowledge-based reasoning by integrating a random forest (RF) predictive model with a failure modes, effects, and criticality analysis (FMECA) knowledge graph (KG). This hybrid architecture aims to address a persistent challenge in engineering maintenance: Bridging the gap between predictive analytics and actionable decision support.

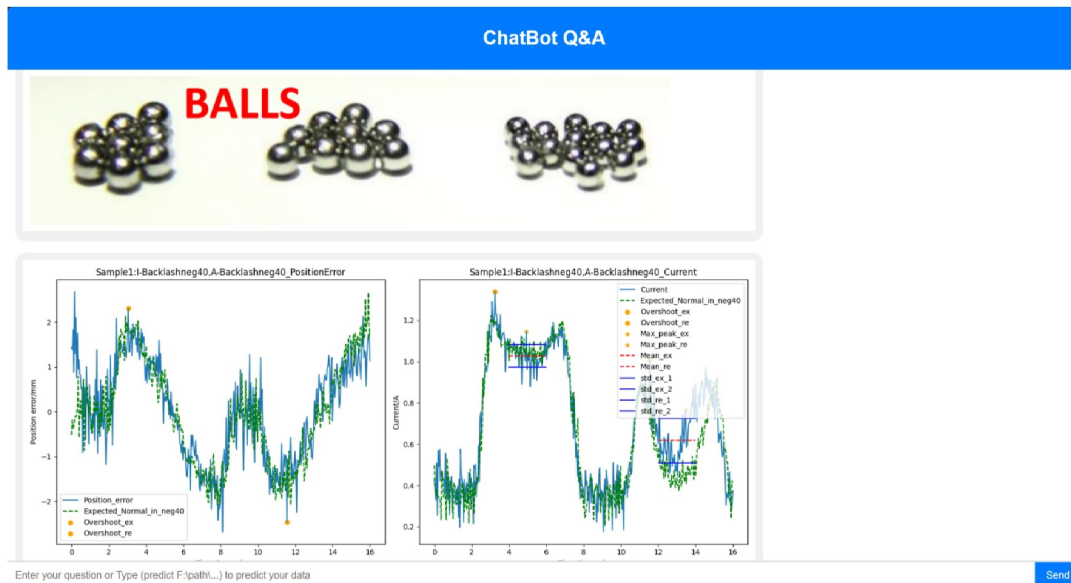


FIGURE 26 | Advisor interaction (part 2).

The main contributions of this work can be summarised as follows. First, a linear actuator was adopted as a representative engineered asset to demonstrate how domain-specific FMECA knowledge can be structured and formalised within a graph-based representation using Neo4j. Second, a query-to-knowledge mapping mechanism was developed to enable natural language interaction with the FMECA-KG. This mechanism leverages rule-based text segmentation via the Aho–Corasick algorithm to associate user queries with relevant fault knowledge. Third, an RF-based fault diagnosis model was trained using operational data, enabling reliable discrimination between normal operation and multiple degradation modes, including backlash, spalling and insufficient lubrication. Finally, these components were integrated into a unified QA system, allowing users to obtain prescriptive maintenance guidance either through natural language queries or by uploading measurement data.

From an AI perspective, the study demonstrates the feasibility and value of coupling machine learning-based inference with explicit symbolic knowledge representations for engineering decision-making. The results suggest that such hybrid AI architectures can enhance interpretability, user interaction and practical utility in maintenance applications. More broadly, the proposed framework contributes to ongoing research in AI for engineering by illustrating how predictive models and structured domain knowledge can be combined to support intelligent maintenance systems.

4.2 | Limitations

4.2.1 | Limitations of FMECA Knowledge Graph

In practice, the FMECA is performed through manual data collection, data entry and data editing. The proposed method is only applicable to a small-scale FMECA when the assets involved are relatively simple. For complex assets, a large-scale FMECA may significantly increase labour and time expenditure. Similarly, defining attributes and relationships in

the FMECA also incurs considerable cost. As the scale of the FMECA expands, the number and complexity of entity types, as well as the relationships between entities, increase substantially. The above factors lead to inherent limitations in the scale and complexity of a manually constructed KG, which consequently constrain the ability of the QA system to comprehend the knowledge layer.

4.2.2 | Limitations of Natural Language Processing (NLP)

The natural language interaction layer in this study relies on rule-based and dictionary-driven NLP techniques. Although effective for constrained query types, this approach cannot capture semantic nuance and contextual dependencies. Queries containing previously unseen vocabulary or more complex linguistic structures may therefore be interpreted incorrectly. These limitations highlight the trade-off between transparency and expressive power in NLP system design, and indicate that the current solution is not readily scalable to open-ended or large domain knowledge interaction scenarios.

4.3 | Future Work

4.3.1 | Knowledge Graph and Ontology

To further enrich the content of the FMECA-KG and expand its structure, constructing the KG from an ontological perspective is a considerable task. Ontology, as a powerful tool for knowledge organisation, can store rich knowledge representations and reasoning logic. For the fault analysis of engineered assets, an ontology can provide conceptual definitions for all entities, ranging from components to specific fault types. Enterprises can progressively enhance the knowledge base by constructing an ontology and establishing a set of knowledge-reasoning mechanisms. Through the ontology, the KG can automatically import all relevant content, eliminating the associated processing costs.

4.3.2 | Natural Language and Deep Learning

Deep learning techniques can be used to classify data features, and can also serve as powerful tools for natural language processing (NLP), such as pre-trained language models (PLMs). To this end, both a large corpus training dataset and an extensive knowledge base are necessary. Fault analysis is a specialised domain, and the analytical approach also varies depending on the target assets. How PLMs can be integrated, and whether the structure of a new larger knowledge base shares similarities with the FMECA-KG, will require further investigation.

4.3.3 | Towards Intelligent Planning in Maintenance Systems

Although the proposed system supports prescriptive recommendations, the generation of complete maintenance plans remains largely dependent on human expertise. A promising direction for future research is to combine knowledge-based reasoning, machine learning and planning algorithms to enable automated or semi-automated maintenance planning. Such capabilities would represent a significant step towards intelligent closed-loop maintenance systems. However, transferring and adapting maintenance knowledge across diverse industrial contexts remains a challenging largely unexplored problem in AI for engineering.

Author Contributions

Hongyi Lin: investigation, methodology, software, formal analysis, data curation, visualization, writing – review and editing, writing – original draft, conceptualization, validation. **Agusmian Partogi Ompusunggu:** conceptualization, methodology, writing – review and editing, project administration, supervision, resources.

Funding

The authors have nothing to report.

Conflicts of Interest

The authors declare no conflicts of interest.

Data Availability Statement

The data that support the findings of this study are openly available in Cranfield University at <https://doi.org/10.17862/cranfield.rd.5097649>.

References

1. D. K. Moulla, E. Mnkandla, M. Aoubakar, A. A. Adamou, and A. Abran, “PdM-FSA: Predictive Maintenance Framework With Fault Severity Awareness in Industry 4.0 Using Machine Learning,” *International Journal of Electrical and Computer Engineering* 14, no. 6 (2024): 7211–7223, <https://doi.org/10.11591/ijece.v14i6.pp7211-7223>.
2. A. P. Ompusunggu, J. M. Papy, and S. Vandenplas, “Kalman-Filtering-Based Prognostics for Automatic Transmission Clutches,” *IEEE/ASME Transactions on Mechatronics* 21, no. 1 (2016): 419–430, <https://doi.org/10.1109/TMECH.2015.2440331>.
3. Y. Wei, Y. Wang, S. Lu, X. Li, and H. Su, “Adaptive Integration of Closed-Loop Manufacturing Scheduling and Predictive Maintenance Based on RUL Prediction by CNN-SVR Model,” *Expert Systems with Applications* 311 (2026): 131269, <https://doi.org/10.1016/j.eswa.2026.131269>.

4. F. Maulana, A. Starr, and A. P. Ompusunggu, “Explainable Data-Driven Method Combined With Bayesian Filtering for Remaining Useful Lifetime Prediction of Aircraft Engines Using NASA CMAPSS Datasets,” *Machines* 11, no. 2 (2023): 163, <https://doi.org/10.3390/machines11020163>.
5. M. Y. You and G. Meng, “A Modularized Framework for Predictive Maintenance Scheduling,” *Proceedings of the Institution of Mechanical Engineers—Part O: Journal of Risk and Reliability* 226, no. 4 (2012): 380–391, <https://doi.org/10.1177/1748006x11431209>.
6. G. Tod, A. P. Ompusunggu, and E. Hostens, “An Improved First-Principle Model of AC Powered Solenoid Operated Valves for Maintenance Applications,” *ISA Transactions* 135 (2023): 551–566, <https://doi.org/10.1016/j.isatra.2022.09.032>.
7. R. Brijder, S. Helsen, and A. P. Ompusunggu, “Switching Kalman Filtering-Based Corrosion Detection and Prognostics for Offshore Wind-Turbine Structures,” *Wind* 3, no. 1 (2023): 1–13, <https://doi.org/10.3390/wind3010001>.
8. S. Vásquez, J. Verhelst, R. Brijder, and A. P. Ompusunggu, “Detection, Prognosis and Decision Support Tool for Offshore Wind Turbine Structures,” *Wind* 2, no. 4 (2022): 747–765, <https://doi.org/10.3390/wind2040039>.
9. O. O. Aremu, D. Hyland-Wood, and P. R. McAree, “A Relative Entropy Weibull-SAX Framework for Health Indices Construction and Health Stage Division in Degradation Modeling of Multivariate Time Series Asset Data,” *Advanced Engineering Informatics* 40 (2019): 121–134, <https://doi.org/10.1016/j.aei.2019.03.003>.
10. R. Sipos, D. Fradkin, F. Moerchen, and Z. Wang, “Log-Based Predictive Maintenance,” in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Association for Computing Machinery, 2014), 1867–1876, <https://doi.org/10.1145/2623330.2623340>.
11. T. C. Kister and B. Hawkins, *Maintenance Planning and Scheduling: Streamline Your Organization for a Lean Environment* (Elsevier, 2006).
12. S. Arena, E. Florian, I. Zennaro, P. F. Orrù, and F. Sgarbossa, “A Novel Decision Support System for Managing Predictive Maintenance Strategies Based on Machine Learning Approaches,” *Safety Science* 146 (2022): 105529, <https://doi.org/10.1016/j.ssci.2021.105529>.
13. IEC, IEC 60812, Failure Modes and Effects Analysis (FMEA and FMECA), (Tech. Rep., IEC, 2018).
14. V. K. Chaudhri, N. Chittar, and M. Genesereth, An Introduction to Knowledge Graphs, (Tech. Rep., Stanford University, 2021).
15. N. Francis, A. Green, P. Guagliardo, et al., “Cypher: An Evolving Query Language for Property Graphs,” in *Proceedings of the 2018 International Conference on Management of Data* (Association for Computing Machinery, 2018), 1433–1445, <https://doi.org/10.1145/3183713.3190657>.
16. G. Chen, G. Zeng, Y. Wang, X. Wang, X. Xie, and Z. Yang, “Risk Analysis Method of a Certain Type of Power Plant Based on KG-KGCN,” *Hangkong Dongli Xuebao/Journal of Aerospace Power*. 38, no. 10 (2023): 2516–2526, <https://doi.org/10.13224/j.cnki.jasp.20220007>.
17. D. Zhang, Y. Zhang, G. Zhang, and C. Qian, “An Exploratory Architecture Using Data and Knowledge to Close the Loop Between Design and Maintenance From a Product Quality Perspective,” *Advanced Engineering Informatics* 56 (2023): 101920, <https://doi.org/10.1016/j.aei.2023.101920>.
18. M. I. Jordan and T. M. Mitchell, “Machine Learning: Trends, Perspectives, and Prospects,” *Science*. 349, no. 6245 (2015): 255–260, <https://doi.org/10.1126/science.aaa8415>.
19. A. Bey-Temsamani, M. Engels, A. Motten, S. Vandenplas, and A. P. Ompusunggu, “A Practical Approach to Combine Data Mining and Prognostics for Improved Predictive Maintenance,” in *The Data Mining*

20. P. Khazaelpour and S. H. Zolfani, “FUCOM-Optimization Based Predictive Maintenance Strategy Using Expert Elicitation and Artificial Neural Network,” *Expert Systems with Applications* 238 (2024): 121322, <https://doi.org/10.1016/j.eswa.2023.121322>.
21. C. L. Liu, B. Xiao, and S. S. Hsu, “Self-Supervised Learning for Remaining Useful Life Prediction Using Simple Triplet Networks,” *Advanced Engineering Informatics* 64 (2025): 103038, <https://doi.org/10.1016/j.aei.2024.103038>.
22. C. Chen, Y. Liu, S. Wang, et al., “Predictive Maintenance Using Cox Proportional Hazard Deep Learning,” *Advanced Engineering Informatics* 44 (2020): 101054, <https://doi.org/10.1016/j.aei.2020.101054>.
23. J. Chatterjee and N. Dethlefs, “Automated Question-Answering for Interactive Decision Support in Operations & Maintenance of Wind Turbines,” *IEEE Access* 10 (2022): 84710–84737, <https://doi.org/10.1109/ACCESS.2022.3197167>.
24. L. Kohl, S. Eschenbacher, P. Besinger, and F. Ansari, “Large Language Model-Based Chatbot for Improving Human-Centricity in Maintenance Planning and Operations,” in *PHM Society European Conference Vol 8* (2024), 12.
25. D. Yang, H. Wang, S. Shao, and S. Liu, “Knowledge-Enhanced Large Language Model-Based Assistance Training System for Subway Maintenance Personnel,” in *Proceedings of the 5th International Conference on Artificial Intelligence and Computer Engineering* (2024): 25–29.
26. X. Cao, W. Xu, J. Zhao, Y. Duan, and X. Yang, “Research on Large Language Model for Coal Mine Equipment Maintenance Based on Multi-Source Text,” *Applied Sciences (Switzerland)* 14, no. 7 (2024): 2946, <https://doi.org/10.3390/app14072946>.
27. Y. Li, R. Kong, Z. Lyu, et al., “FlexSpec: Frozen Drafts Meet Evolving Targets in Edge-Cloud Collaborative LLM Speculative Decoding,” preprint, arXiv:2601.00644 (2026).
28. D. Zhang, W. Kang, W. Zhao, et al., “The Research on Synchronous Fault Diagnosis Method for Camera Based on Knowledge Graph and Large Language Model,” in *IEEE* (2025), 741–746.
29. C. Ruiz-Carcel and A. Starr, “Data-Based Detection and Diagnosis of Faults in Linear Actuators,” *IEEE Transactions on Instrumentation and Measurement* 67, no. 9 (2018): 2035–2047, <https://doi.org/10.1109/tim.2018.2814067>.
30. A. Liaw and M. Wiener, “Classification and Regression by Randomforest,” *R News* 2, no. 3 (2002): 18–22, <https://journal.r-project.org/articles/RN-2002-022/RN-2002-022.pdf>.
31. R. Kizito, P. Scruggs, X. Li, R. Kress, M. Devinney, and T. Berg, “The Application of Random Forest to Predictive Maintenance,” in *Institute of Industrial and Systems Engineers (IISE)* (2018), 354–359.
32. H. A. Salman, A. Kalakech, and A. Steiti, “Random Forest Algorithm Overview,” *Babylonian Journal of Machine Learning* 2024 (2024): 69–79, <https://doi.org/10.58496/bjml/2024/007>.
33. Y. Wan, D. Liu, and J. C. Ren, “A Modeling Method of Wide Random Forest Multi-Output Soft Sensor With Attention Mechanism for Quality Prediction of Complex Industrial Processes,” *Advanced Engineering Informatics* 59 (2024): 102255, <https://doi.org/10.1016/j.aei.2023.102255>.
34. S. Wu, Y. Hu, L. Zhang, S. Liu, R. Xie, and Z. Yin, “Intelligent Risk Identification for Drilling Lost Circulation Incidents Using Data-Driven Machine Learning,” *Reliability Engineering & System Safety* (2024): 252, <https://doi.org/10.1016/j.res.2024.110407>.
35. J. Moćkus, *On Bayesian Methods for Seeking the Extremum* (Springer, 1974), 400–404.
36. C. Bian, Q. Liu, X. Zhang, et al., “An Efficient Mixed Constrained Bayesian Optimization for Handling Known and Unknown Constraints,” *Advanced Engineering Informatics* 62 (2024): 102704, <https://doi.org/10.1016/j.aei.2024.102704>.
37. J. Snoek, H. Larochelle, and R. P. Adams, “Practical Bayesian Optimization of Machine Learning Algorithms,” in *Advances in Neural Information Processing Systems* (2012), 2951–2959.
38. J. Wu, X. Y. Chen, H. Zhang, L. D. Xiong, H. Lei, and S. H. Deng, “Hyperparameter Optimization for Machine Learning Models Based on Bayesian Optimization,” *Journal of Electronic Science and Technology* 17, no. 1 (2019): 26–40, <https://doi.org/10.11989/JEST.1674-862X.80904120>.
39. X. Yu, C. Jing, W. Shen, and M. Liu, “A Multi-Stage Bayesian Optimization Algorithm for Physical Parameter Inversion in Overlying Strata of Mined-Out Areas,” *Computers and Geotechnics* 191 (2026): 107815, <https://doi.org/10.1016/j.compgeo.2025.107815>.
40. A. S. Raihan and I. Ahmed, “Guiding the Sequential Experiments in Autonomous Experimentation Platforms Through EI-Based Bayesian Optimization and Bayesian Model Averaging,” preprint, arXiv:2302.13360 (2023).
41. K. Yang, M. Affenzeller, and G. Dong, “A Parallel Technique for Multi-Objective Bayesian Global Optimization: Using a Batch Selection of Probability of Improvement,” *Swarm and Evolutionary Computation* 75 (2022): 101183, <https://doi.org/10.1016/j.swevo.2022.101183>.
42. T. L. M. Suryanto, A. P. Wibawa, and N. A. Hariyono, “Comparative Performance of Transformer Models for Cultural Heritage in NLP Tasks,” *Advance Sustainable Science, Engineering and Technology* 7, no. 1 (2025): 0250115, <https://doi.org/10.26877/asset.v7i1.1211>.
43. H. Liu, “QA Based on Medical Knowledge Graph,” *GitHub* (2018), <https://github.com/tedrepo/QABasedOnMedicalKnowledgeGraph>.
44. C. H. Lin and S. C. Chang, “Efficient Pattern Matching Algorithm for Memory Architecture,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 19, no. 1 (2011): 33–41, <https://doi.org/10.1109/tvlsi.2009.2028346>.



DSPACE

<https://dspace.org/>

Intelligent advisor system for prescriptive maintenance of engineered assets using failure modes, effects and criticality analysis, knowledge graph and machine learning

Lin, Hongyi; Ompusunggu, Agusmian Partogi

2026-12-31

Wiley

<https://doi.org/10.1049/aie2.70019>

Lin H, Ompusunggu AP. (2026) Intelligent advisor system for prescriptive maintenance of engineered assets using failure modes, effects and criticality analysis, knowledge graph and machine learning. Artificial Intelligence for Engineering, Available online 21 May 2026