A Neural-Symbolic Network for Interpretable Fault Diagnosis of Rolling Element Bearings Based on Temporal Logic

Ruoyao Tian, Mengqian Cui, and Gang Chen, *Member, IEEE*

***Abstract*— This study examines the issue of interpretability in fault diagnosis for rolling bearings using a symbolic learning technique. We propose the adoption of weighted signal temporal logic (wSTL) as a formal language and introduce the temporal logic network (TLN) as a neural-symbolic learning architecture capable of encoding symbolic wSTL representations for input signals. TLN comprises three subnetworks: a basic predicate network for abstracting features and generating predicates from vibration signals, an autoencoder for identifying significant signal components, and a logic network for constructing a formal language that aids in fault classification and model interpretation. To improve comprehensibility, timed failure propagation graphs (TFPGs) are used to visually represent the logical relationships and propagation of fault events. Experimental results demonstrate TLN's ability to extract impulse fault patterns from signals, accurately describe fault events through learned wSTL formulas, and enhance understanding of fault events for nonexpert individuals through TFPGs. These findings contribute to the field of fault diagnosis in rolling bearings by incorporating symbolic learning techniques, using formal language representation and TFPG for improved interpretability.**

***Index Terms*— Interpretable fault diagnosis, rolling element bearing, signal temporal logic (STL), symbolic learning, temporal logic network (TLN).**

I. Introduction

MODERN machines are becoming increasingly sophisticated with intricate components and intricate interdependencies. The complexity poses significant challenges when it comes to identifying and addressing faults that may arise during their operation [1]. Within these machines, bearings serve as critical components that facilitate smooth and efficient operation. However, the occurrence of faults in bearings can lead to catastrophic consequences, including unexpected downtime, expensive repairs, and potentially hazardous situations [2]. Therefore, accurate and timely fault diagnosis of bearings is crucial to ensure the safe and reliable functioning of machines.

Traditional fault diagnosis methods, such as time-domain [3], frequency-domain [4], and wavelet-based analysis [5], have long been used to identify and classify bearing faults. However, as the complexity of machinery and systems increases, the traditional fault diagnosis methods struggle to provide accurate diagnosis results. On the other hand, machine learning (ML) models have shown great potential in fault diagnosis due to their ability to recognize complex patterns in data [6]. However, their lack of interpretability has been a significant challenge in deploying them in critical applications [7]. This limitation poses challenges when it comes to making informed decisions regarding maintenance, repairs, and replacement of key components, since they do not provide explanations about *how and why* they make a decision and reveal the fault mechanism. As a result, interpretable fault diagnosis has emerged as a crucial field that aims to address this challenge by providing transparent and interpretable fault diagnosis results.

Several interpretable ML methods have been developed to bridge the gap between accuracy and explainability, enabling experts and nonexperts to comprehend and interpret the diagnostic results effectively [8], [9], [10]. These methods can be roughly classified into three classes: visual explanations, feature-relevance explanations, and knowledge-extraction explanations, respectively. Visual explanations use graphical representations to illustrate a model's decision-making process, using techniques such as saliency maps [11], class activation maps [12], and feature visualization [13]. Feature-relevance explanations aim to clarify how individual input features influence the model's output, using approaches such as feature importance [14], partial dependence plots (PDPs) [15], local interpretable model-agnostic explanations (LIMEs) [16], and shapley additive explanations (SHAPs) [17]. Knowledge-extraction explanations strive to demystify the inner workings of complex models by converting their outputs into more understandable formats such as rules or tree structures, using techniques such as decision trees [18], rule extraction [19], model distillation [20], and prototype selection [21].

However, these interpretive methods face significant limitations, especially in time-series fault diagnosis. Visual explanations, typically effective for image-based analysis, struggle to capture the evolving nature of time-series faults.

Manuscript received 2 December 2023; revised 14 January 2024; accepted 30 January 2024. Date of publication 4 March 2024; date of current version 27 March 2024. This work was supported in part by the National Key Research and Development Program of China under Grant 2021YFB3301402, in part by the National Natural Science Foundation of China under Grant 52305105, and in part by the Basic and Applied Basic Research Foundation of Guangdong Province under Grant 2022A1515240027 and Grant 2023A1515010812. The Associate Editor coordinating the review process was Dr. Siliang Lu. *(Corresponding author: Gang Chen.)*

Ruoyao Tian and Gang Chen are with the Shien-Ming Wu School of Intelligent Engineering, South China University of Technology, Guangzhou, Guangdong 511442, China (e-mail: 1502524138@qq.com; gangchen@scut.edu.cn).

Mengqian Cui is with Guangdong Rural Credit Union, Guangdong 510627, China (e-mail: 2201156119@qq.com).

Digital Object Identifier 10.1109/TIM.2024.3373103

Feature-relevance explanations fail to address the temporal interdependencies and patterns crucial in time-series data. Knowledge-extraction explanations often oversimplify complex data, losing essential information needed for accurate time-series fault diagnosis.

To address the above limitations, this article proposed a temporal logic explanation framework for bearing fault diagnosis, which uses temporal logic formulas to explain the dynamic properties of time-series signals. Temporal logic explanations represent a sophisticated interpretive method in ML, particularly tailored for models dealing with sequential or time-series data. The imperative for studying temporal logic explanations becomes particularly pronounced in the realm of machinery fault diagnosis, where time-series data play a pivotal role. In such applications, an interpretable logic network designed for inferring temporal logic explanations offers marked advantages over traditional deep learning models or methods based on formal reasoning. Unlike these conventional approaches, an interpretable logic network adeptly captures and represents the intricate temporal patterns and dependencies critical in machinery fault diagnosis. For example, in the monitoring of rotating machinery, the temporal evolution of vibration signatures can be key to early fault detection and prevention. A deep learning model might accurately predict a fault, but without the clarity provided by temporal logic explanations, the underlying temporal sequence leading to this fault remains obscured. An interpretable logic network, in contrast, provides transparent insights into how specific temporal patterns correlate with certain types of faults, enhancing both the understanding and trust in the predictive system. This transparency is not just crucial for accurate diagnostics but also enables more targeted and effective maintenance strategies, thereby reducing downtime and costs. Thus, in the context of machinery fault diagnosis, the study and application of interpretable logic networks for temporal logic explanation are not only academically intriguing but also carry substantial practical significance, offering a more profound and actionable understanding of time-dependent fault dynamics.

Current temporal logic explanations in ML, while effective in certain scenarios, face significant challenges in addressing the complex, time-dependent dynamics of sequential data. These methods struggle to map intricate temporal relationships and dependencies onto formal logic structures in a way that is both accurate and user-friendly. The core issue lies in developing a framework that can interpret the dynamic interplay of features over time, balancing the fidelity of the explanation with its clarity and accessibility to users.

To overcome these challenges, this article introduces a novel neural-symbolic network (NSN) architecture, named the temporal logic network (TLN), designed specifically for rolling element-bearing fault diagnosis. The TLN addresses key limitations of current temporal logic methods by enhancing the interpretability and applicability of formal languages in complex machinery systems. The existing temporal logic-based fault diagnosis methods, while successful in certain monitoring tasks [22], often rely on predetermined structures or predefined atomic words, limiting their effectiveness in modern, complex devices [23]. Furthermore, when these methods use neural networks for learning formal languages, they typically focus on fault diagnosis performance, leading to a lack of transparency in the learning process and a failure to guarantee formal performance [24].

The TLN incorporates the weighted signal temporal logic (wSTL), which enhances the representation of vibration signal features with embedded signal processing techniques and offers differentiable quantitative semantics. This advancement allows for a more precise capture and interpretation of temporal features relevant to fault diagnosis. The integration of wSTL within deep learning models in the TLN framework represents a significant step forward. The parameters of the modules in TLN can be translated into wSTL formulas, enabling a clearer interpretation of fault events. The modular design of the TLN also allows for flexible adjustments in the framework, catering to various diagnostic scenarios.

Moreover, the TLN uses a temporal fault propagation graph (TFPG) to visualize the relationships among fault events. This visualization, combined with the formal wSTL formulas, provides a comprehensive understanding of fault dynamics, making it accessible to individuals without specialized knowledge. This approach not only enhances the interpretability of the diagnostic process but also links underlying features to their respective generation mechanisms clearly and logically. In comparison to the state-of-the-art methods, the contributions of this approach can be summarized as follows.

1) Based on signal temporal logic (STL), we propose wSTL, which can capture the features of vibration signals with signal processing techniques embedded and has a differentiable quantitative semantic.
2) We embed wSTL in deep learning models and propose an NSN architecture. The parameters of the modules in TLN can be transferred to a wSTL formula for fault event interpretation. The modular design of TLN allows us to change the framework's arrangement for flexible formula creation.
3) Using a TFPG, we visualize the relationships among fault events. By combining these visualizations with formal formulas, individuals without specialized knowledge can comprehend the underlying features and link them to their respective generation mechanisms.

The organization of the rest of this article is as follows. Section II provides a comprehensive definition of formal languages and TFPG. Vital modules are presented in Section III. The architecture of TLN is showcased in Section IV. Section V uses three datasets to assess the TLN's interpretability and the effectiveness of the visualization techniques, and then highlights the advantages of our method by comparing it with the existing models. Finally, Section VI summarizes the findings and draws the conclusions.

II. Preliminaries and Notations

In this section, we introduce the definitions of formal language and TFPG, respectively. The first part presents wSTL's syntax and semantics. The second part introduces the formal definition of TFPG and its application as a visualization tool.

A. Weighted Signal Temporal Logic

Definition 1 (Signal): In signal theory, a signal is a mapping from the time domain to the signal space. Given sets

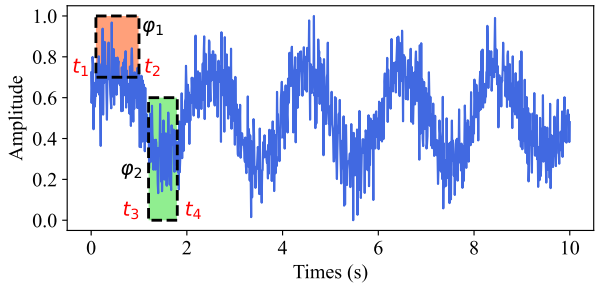


Fig. 1. Time series $f(x)$. The regions framed by the black dotted box are where the fault characteristic may be present. The red areas are where the signal may reach and the green areas are where the signal must reach.

$\mathbb{A}$ and $\mathbb{B}$, the set of mappings from $\mathbb{A}$ to $\mathbb{B}$ can be denoted by $\mathscr{F}(\mathbb{A}, \mathbb{B})$. Let $\mathbb{D} = \{k\tau_0 \mid k \in \mathbb{N}^+\}$ represent the discrete time domain, with τ_0 as the sampling interval, and $\mathbb{R}^n$ as the n-dimensional signal space. Thus, a signal $x \in \mathscr{F}(\mathbb{D}, \mathbb{R}^n)$. For this article, x is a discrete-time signal, with $x_i[t]$ $(i = 1, 2, \ldots, n)$ denoting the amplitude of the ith dimension of x at time t.

Definition 2 (STL): STL is one type of temporal logic that describes the time-domain characteristics of discrete signals. Its syntax can be defined recursively as

$$\varphi := \mu \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid F_{[\tau_1,\tau_2]}\varphi \mid G_{[\tau_1,\tau_2]}\varphi \tag{1}$$

where μ is a predicate logic with the form $f(x) \sim c$. $f(\cdot) \in \mathscr{F}(\mathbb{R}^n, \mathbb{R})$ is a function that maps the n-dimensional signal x to a real number; $\sim \in \{<, \geq\}$ denotes the comparison operator; and $c \in \mathbb{R}$ is a constant. $\neg\varphi$ is the negation of φ. $\wedge$ and $\vee$ are Boolean operators that represent logical AND and logical OR, respectively. $[\tau_1, \tau_2]$ denotes a time interval with $\tau_1 \leq \tau_2$ and $\tau_1, \tau_2 \in \mathbb{N}^+$. F and G are temporal operators that usually appear before φ. $F_{[\tau_1,\tau_2]}\varphi$ means that φ is true "at least once" within $[\tau_1, \tau_2]$, and $G_{[\tau_1,\tau_2]}\varphi$ means that φ is "always" true within $[\tau_1, \tau_2]$.

According to Definition 2, STL can describe the property of the discrete signal shown in Fig. 1.

Example 1: Fig. 1 illustrates a discrete time series x, and the signal features covered by the red region and the green region can be represented by STL formulas

$$\varphi_1 = F_{[t_1,t_2]} f(x) \geq 0.7 \tag{2}$$

$$\varphi_2 = G_{[t_3,t_4]} f(x) < 0.6. \tag{3}$$

Equation (2) indicates that $f(x)$ is greater than or equal to 0.7 at least once within $[t_1, t_2]$; (3) indicates that $f(x)$ is always less than 0.6 within $[t_3, t_4]$. In the rest of this article, signal features are also referred to as fault events, and an STL formula is the formal language representation of the fault events.

Definition 3 (Robustness Degree of STL): STL is equipped with quantitative semantics [24], called robustness degree $\rho(x, \varphi, t_0)$. It measures the degree of how much the signal x satisfies the STL formula φ at time t_0 and is equivalent to the ability of φ to describe the behavior of x at t_0. If we denote the set of STL formulas by Ψ, then robustness degree $\rho : \mathscr{F}(\mathbb{D}, \mathbb{R}^n) \times \Psi \to \mathbb{R}$ is a function that maps a signal x and an STL formula φ to a real number. The robustness degree $\rho(x, \varphi, t_0)$ of STL is defined as follows:

$$\begin{aligned}
\rho(x, f(x) < c, t_0) &= c - f(x) \\
\rho(x, f(x) \geq c, t_0) &= f(x) - c \\
\rho(x, \neg\varphi, t_0) &= -\rho(x, \varphi, t_0) \\
\rho(x, \varphi_1 \wedge \varphi_2, t_0) &= \min(\rho(x, \varphi_1, t_0), \rho(x, \varphi_2, t_0)) \\
\rho(x, \varphi_1 \vee \varphi_2, t_0) &= \max(\rho(x, \varphi_1, t_0), \rho(x, \varphi_2, t_0)) \\
\rho\left(x, G_{[\tau_1,\tau_2]}\varphi, t_0\right) &= \min_{t' \in [t_0+\tau_1, t_0+\tau_2]} \rho\left(x, \varphi, t'\right) \\
\rho\left(x, F_{[\tau_1,\tau_2]}\varphi, t_0\right) &= \max_{t' \in [t_0+\tau_1, t_0+\tau_2]} \rho\left(x, \varphi, t'\right)
\end{aligned} \tag{4}$$

where t_0 in $\rho(x, \varphi, t_0)$ means that the robustness degree is computed from t_0, and $\rho(x, \varphi, t_0)$ is also written as $\rho(x, \varphi)$ when $t_0 = 0$. If $\rho(x, \varphi, t_0) \geq 0$, the behavior of x satisfies the description of φ and is denoted as $x \models \varphi$; if $\rho(x, \varphi, t_0) < 0$, the behavior of x does not satisfy the description of φ and is denoted as $x \not\models \varphi$.

However, STL has some drawbacks.

1) A formula with the form $F_{[\tau_1,\tau_2]}\mu$ or $G_{[\tau_1,\tau_2]}\mu$ is called an atomic formula. In Definition 1, all the atomic formulas within an STL formula φ have the same weight, which makes it impossible to distinguish the contribution of different atomic formulas to φ.
2) The robustness degree of STL is not derivable because of the presence of min and max operators. If we embed STL in a neural network, the training process will likely be ineffective.

Definition 4 (wSTL): The syntax of wSTL is an extension of STL, which can be defined recursively as

$$\begin{aligned}
\varphi^w := \mu \mid \neg\varphi^w \mid \varphi_1^w \wedge \varphi_2^w \mid \varphi_1^w \vee \varphi_2^w \\
\mid F_{[\tau_1,\tau_2]}\varphi^w \mid G_{[\tau_1,\tau_2]}\varphi^w \mid \varphi_1^w \mathcal{U}_{[\tau_1,\tau_2]}\varphi_2^w.
\end{aligned} \tag{5}$$

The semantics of the predicate μ, the operators $\wedge$, $\vee$, $\neg$, and the temporal operators F and G are the same as in Definition 1. There are two differences between wSTL and STL

1) Each formula φ_i in wSTL has a weight w_i, which represents the contribution of φ_i to φ.
2) A new temporal operator $\mathcal{U}$ is added to wSTL. $\varphi_1^w \mathcal{U}_{[\tau_1,\tau_2]} \varphi_2^w$ means that after satisfying the description of φ_1^w, the behavior of the signal will satisfy the description of φ_2^w within $[\tau_1, \tau_2]$. $\mathcal{U}$ enables wSTL for characterizing the propagation of signal features in the time domain.

Definition 5 (Robustness Degree of wSTL): Like STL, wSTL is also equipped with quantized semantics called weighted robustness degree. Given an n-dimensional signal x and a wSTL formula φ, the weighted robustness degree $\rho^w(x, \varphi, t_0)$ is defined as follows:

$$\begin{aligned}
\rho^w(x, f(x) < c, t_0) &= c - f(x) \\
\rho^w(x, f(x) \geq c, t_0) &= f(x) - c \\
\rho^w\left(x, G_{[\tau_1,\tau_2]}\varphi^w, t_0\right) &= g^G\left(w, \left[\rho^w\left(x, \varphi, t'\right)\right]_{t' \in [t_0+\tau_1, t_0+\tau_2]}\right) \\
\rho^w\left(x, F_{[\tau_1,\tau_2]}\varphi^w, t_0\right) &= g^F\left(w, \left[\rho^w\left(x, \varphi, t'\right)\right]_{t' \in [t_0+\tau_1, t_0+\tau_2]}\right) \\
\rho^w\left(x, \wedge_{i=1}^N \varphi_i^{w_i}, t_0\right) &= g^\wedge\left(\left[w_i, \rho^w(x, \varphi_i, t_0)\right]_{i=1,2,\ldots,N}\right) \\
\rho^w\left(x, \vee_{i=1}^N \varphi_i^{w_i}, t_0\right) &= g^\vee\left(\left[w_i, \rho^w(x, \varphi_i, t_0)\right]_{i=1,2,\ldots,N}\right) \\
\rho^w\left(x, \varphi_1^w \mathcal{U}_{[\tau_1,\tau_2]} \varphi_2^w, t_0\right) &= g^{\mathcal{U}}\left(\left[\rho^w\left(x, \varphi_2^w, t'\right)_{t' \in [t_0+\tau_1, t_0+\tau_2]}\right]\right. \\
&\qquad \left.\left[\rho^w\left(x, \varphi_1^w, t''\right)_{t'' \in [t_0, t']}\right]\right)
\end{aligned} \tag{6}$$

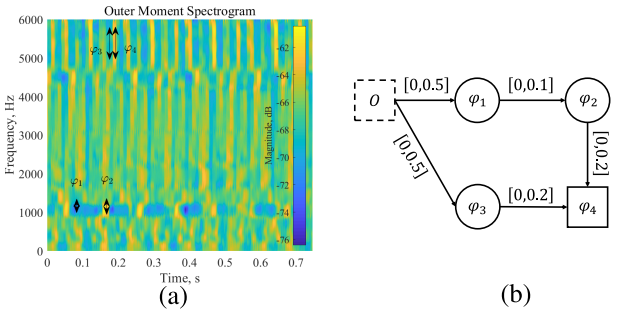
where $g^G(\cdot)$, $g^F(\cdot)$, $g^\wedge(\cdot)$, $g^\vee(\cdot)$, and $g^{\mathcal{U}}(\cdot)$ are the robustness degree computation functions of $G_{[\tau_1,\tau_2]}\varphi^w$, $F_{[\tau_1,\tau_2]}\varphi^w$, $\varphi_1^{w_1} \wedge \varphi_2^{w_2} \wedge, \ldots, \wedge \varphi_N^{w_N}$, $\varphi_1^{w_1} \vee \varphi_2^{w_2} \vee, \ldots, \vee \varphi_N^{w_N}$, and $\varphi_1^w \mathcal{U}_{[\tau_1,\tau_2]} \varphi_2^w$, respectively. The specific mathematical expressions are defined as in (7), shown at the bottom of the page.

B. Timed Failure Propagation Graph

Definition 6 (TFPG): TFPG is a visualization model adopted by Vanderbilt University for system failure analysis. Formally, a TFPG is a tuple $G = \langle F, D, E, ET, dc, DP \rangle$, where: 1) F is a set of failure mode nodes; 2) D is a set of discrepancy nodes; 3) E is a set of directed edges that satisfies $E \subseteq V \times V$, where $V = F \cup D$; 4) $ET : E \to I$ is a mapping from a directed edge $e \in E$ to a time interval $[t_{\min(e)}, t_{\max(e)}] \in I$; 5) $dc : D \to \{\text{AND}, \text{OR}\}$ is a mapping from a discrepancy node $d \in D$ to its discrepancy type; and 6) DP maps a discrepancy node $d \in D$ to an atomic formula $\varphi \in \Phi$.

A formal formula can be mapped to a TFPG, enabling us to effectively depict the intricate interconnections among fault occurrences. By examining Example 2, we can acquire a deeper comprehension of the practical implementation of TFPGs in producing interpretable fault diagnosis.

Example 2: Fig. 2(a) illustrates the moment spectrogram of a vibration signal, and Fig. 2(b) is the TFPG describing its characteristics. TFPG is a directed graph, the dashed node is the fault mode node, and the solid nodes are discrepancy nodes, which can indicate fault events and their logical relationships. The solid circle indicates OR, and the solid square indicates AND. The symbol inside the node is the name of the fault event, and the numbers above the directed edge represent the time interval of failure propagation. TFPGs not only describe the logical relationships and propagation between fault events in graphical form but also can be interpreted in natural language. The natural language description of the TFPG shown in Fig. 2(b) is: "The faulty characteristics of the outer ring fault mode are: within 0.5 s after the bearing begins to rotate, fault events φ_1 and φ_3 will occur; within 0.1 s after φ_1 occurs, φ_2 will occur; within 0.2 s after the occurrence of φ_2 and within 0.2 s after the occurrence of φ_3, φ_4 will occur." The natural language description of fault events can be referred to Example 1.



(a) (b)

Fig. 2. Visualization of diagnostic results by TFPG. (a) Moment spectrogram from an outer race fault bearing. (b) TFPG mapped by $\varphi_O = F_{[0,0.5]}((\varphi_1 \mathcal{U}_{[0,0.1]} \varphi_2) \wedge \varphi_3) \mathcal{U}_{[0,0.2]} \varphi_4$.

III. Network Modules

In this section, we present the network modules based on wSTL, and all of them are components of TLN.

A. Wavelet Convolution Module

The wavelet convolution module can extract features from vibration signals. As shown in Fig. 3, a group of Laplace wavelets with different parameters, which can be considered as a filter bank, are used to convolve with the input. The parameters of the wavelets are randomly initialized and to be trained. The following is how the vibration signal is handled in this module:

$$y_i = F(x \times w_i) \tag{8}$$

$$\rho^w\left(x, \wedge_{i=1}^N \varphi_i^{w_i}, t\right) := \begin{cases} \sqrt[N]{\Pi_{i=1}^N (1 + w_i \rho^w(x, \varphi_i, t))} - 1, & \forall i \in [1, 2, \ldots, N], \rho^w(x, \varphi_i, t) > 0 \\ \dfrac{1}{N} \displaystyle\sum_{i=1}^N (-w_i \rho^w(x, \varphi_i, t)), & \text{otherwise} \end{cases}$$

$$\rho^w\left(x, \vee_{i=1}^N \varphi_i^{w_i}, t\right) := \begin{cases} \dfrac{1}{N} \displaystyle\sum_{i=1}^N (w_i \rho^w(x, \varphi_i, t)), & \exists i \in [1, 2, \ldots, N], \rho^w(x, \varphi_i, t) > 0 \\ -\sqrt[N]{\Pi_{i=1}^N (1 - w_i \rho^w(x, \varphi_i, t))} + 1, & \text{otherwise} \end{cases}$$

$$\rho^w\left(x, \Box_{[\tau_1,\tau_2]} \varphi^w, t\right) := \begin{cases} \sqrt[\tau_2-\tau_1]{\Pi_{t'=\tau_1}^{\tau_2} (1 + w \rho^w(x, \varphi, t'))} - 1, & \forall t' \in [t + \tau_1, t + \tau_2], \rho^w\left(x, \varphi, t'\right) > 0 \\ \dfrac{1}{\tau_2 - \tau_1} \displaystyle\sum_{t' \in [t+\tau_1, t+\tau_2]} \left(-w \rho^w\left(x, \varphi, t'\right)\right), & \text{otherwise} \end{cases}$$

$$\rho^w\left(x, \Diamond_{[\tau_1,\tau_2]} \varphi^w, t\right) := \begin{cases} \dfrac{1}{\tau_2 - \tau_1} \displaystyle\sum_{t' \in [t+\tau_1, t+\tau_2]} \left(w \rho^w\left(x, \varphi, t'\right)\right), & \exists t' \in [t + \tau_1, t + \tau_2], \rho^w\left(x, \varphi, t'\right) > 0 \\ -\sqrt[\tau_2-\tau_1]{\Pi_{t'=\tau_1}^{\tau_2}} \left(1 - w \rho^w\left(x, \varphi, t'\right) + 1, & \text{otherwise} \right. \end{cases}$$

$$\rho^w\left(x, \varphi_1^w \mathcal{U}_{[\tau_1,\tau_2]} \varphi_2^w, t\right) := \max_{t' \in [t+\tau_1, t+\tau_2]} \left(\min \left(\rho^w\left(x, \varphi_2^w, t'\right), \min_{t'' \in [t, t']} \rho^w\left(x, \varphi_1^w, t''\right) \right) \right) \tag{7}$$

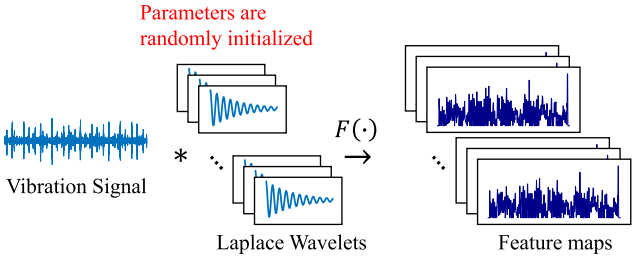


Fig. 3. Computational procedure of the wavelet convolution module. To generate feature maps, the vibration signal is convolved by a set of Laplace wavelets and processed by some other data processing procedures $F(\cdot)$.

where x denotes the vibration signal; w_i denotes the ith Laplace wavelet; y_i denotes the ith feature map; $*$ denotes 1-D convolution operation; and $F(\cdot)$ denotes some other data processing procedures, including batch normalization (BN), nonlinear activation, and max-pooling

$$F(\cdot) = \text{MaxPool1d}(\text{ReLU}(\text{BatchNorm1d}(\cdot))). \tag{9}$$

B. Predicate Generation Module

After getting the features, the predicate generation module can generate some predicates describing them and calculate their robustness degrees. Depending on the comparison operator within the predicate, the module has two modes of computation

$$\mu_{\geq,i} = y_i - c_i, \quad \mu_{<,i} = c_i - y_i \tag{10}$$

where c_i represents the constant within the ith predicate; and $\mu_{\geq,i}$ and $\mu_{<,i}$ represent the robustness degree of $y_i \geq c_i$ and $y_i < c_i$, respectively. Note that c_i is the parameter to be trained in this module.

C. Temporal Logic Modules

In total, there are five temporal logic modules: Temporal_{A1}, Temporal_{O1}, $\text{Temporal}_{\mathcal{U}}$, Temporal_{A2}, and Temporal_{O2}, which are used to construct wSTL formulas. To enhance the clarity of our presentation, we establish the designations of Level I formula, Level II formula, and subsequent levels accordingly. Level I formulas are formed by combining atomic formulas once with $\wedge$, $\vee$, or $\mathcal{U}$, for example, $\varphi_1 \wedge \varphi_2$. Level II formulas are formed by combining Level I formulas with these operators again. Fig. 4 illustrates the computation of these five modules.

According to the syntax of wSTL, a predicate is usually preceded by a temporal operator, and each temporal operator has an effective time interval. We now assume that the effective time intervals have been obtained, the and method for getting them will be presented in Section IV. Temporal_{A1} first adds the temporal operator G before the predicate sequence to generate the atomic formula sequence. Next, this module selects a certain number of atomic formulas and randomly generates a set of weights, combining the atomic formulas using $\wedge$ to form a Level I formula sequence. Finally, ReLU$(\cdot)$ activates the robustness degree sequence, rendering the Level I formulas with positive robustness degrees valid

$$\varphi_j^{w_j} = \rho^w\left(y_i, G_{[\tau_1,\tau_2]_j}\mu_j\right) \tag{11}$$

$$\varphi_{\wedge_I,k}^{w_k} = \text{ReLU}\left(\rho^w\left(y_i, \wedge_{j=1}^{N}\varphi_j^{w_j}\right)\right) \tag{12}$$

where μ_j and $\varphi_j^{w_j}$ denote the robustness degree of the jth predicate and the jth atomic formula, respectively; $[\tau_1, \tau_2]_j$ denotes the jth effective time interval for μ_j; w_j is the weight for φ_j; $\varphi_{\wedge_I,k}^{w_k}$ denotes the kth Level I formula; and N represents the number of atomic formulas that constitute $\varphi_{\wedge_I,k}^{w_k}$. The weights of the formulas are randomly initialized at the beginning and are the parameters to be trained. Similarly, within Temporal_{O1}, the temporal operator F is added before the predicates, and the atomic formulas are connected by $\vee$.

$\text{Temporal}_{\mathcal{U}}$ first adds the temporal operator (F or G) before the predicates and then combines two atomic formulas with $\mathcal{U}$

$$\varphi_{\mathcal{U}_I,k}^{w_k} = \text{ReLU}\left(\rho^w\left(x, \varphi_{j_1}^{w_1}\mathcal{U}_{[0,L]}\varphi_{j_2}^{w_2}\right)\right) \tag{13}$$

where $\varphi_{j_1}^{w_1}$ and $\varphi_{j_2}^{w_2}$ can be any possible atomic formula; and $\varphi_{\mathcal{U}_I,k}^{w_k}$ denotes the kth Level I formula with the form $\varphi_1^w\mathcal{U}_{[0,L]}\varphi_2^w$. To reduce the number of parameters in this module, we fix the propagation time as $[0, L]$.

Level I formulas will be fed into Temporal_{A2} and Temporal_{O2}. Temporal_{A2} randomly generates a set of weights and uses the operator $\wedge$ to combine Level I formulas

$$\varphi_{\wedge_{II},m}^{w_m} = \text{ReLU}\left(\rho^w\left(y_i, \wedge_{k=1}^{M}\varphi_{\wedge_I,k}^{w_k}\right)\right) \tag{14}$$

where w_k represents the weight for $\varphi_{\wedge_I,k}^{w_k}$; $\varphi_{\wedge_{II},m}^{w_m}$ denotes the activated robustness degree for the mth Level II formula; and M represents the number of Level I formulas that constitute $\varphi_{\wedge_{II},m}^{w_m}$. Temporal_{O2} performs the same computation as Temporal_{A2}, with the only difference being that the formulas are connected by $\vee$.

IV. Temporal Logic Network

In this section, we introduce the architecture of TLN. TLN can extract interpretable patterns from vibration signals, identify the time intervals where the patterns are located, and automatically learn a wSTL formula that describes the logical relationship between patterns. As shown in Fig. 5, TLN contains three subnetworks.

1) *Basic Predicate Network:* Extract interpretable patterns from vibration signal and generate a sequence of predicates that describe the patterns.
2) *Autoencoder:* Identify the probability that a sampling point is within an effective time interval.
3) *Logic Network:* Construct wSTL formulas based on effective time intervals and predicate sequence.

Within the basic predicate network, the wavelet convolution module generates 16 pairs of wavelet parameters, which leads to 16 convoluted time-series signals from one vibration signal. Then, the predicate module calculates the robustness degree of the predicates describing these convoluted signals in two cases based on the type of predicates, i.e., $\geq$ and $<$. For a convoluted time series with length N, the output of the basic predicate network $\mu \in \mathbb{R}^{32\times N}$ can be described as follows:

$$\begin{aligned} y_i &= \text{WaveletConv}(x, w_i) \\ \mu_{\geq,i} &= y_i - c_i, \quad \mu_{\leq,i} = c_i - y_i \\ \mu &= \left[\mu_{\geq,0}, \mu_{<,0}, \ldots, \mu_{\geq,i}, \mu_{<,i}, \ldots, \mu_{\geq,15}, \mu_{<,15}\right] \end{aligned} \tag{15}$$

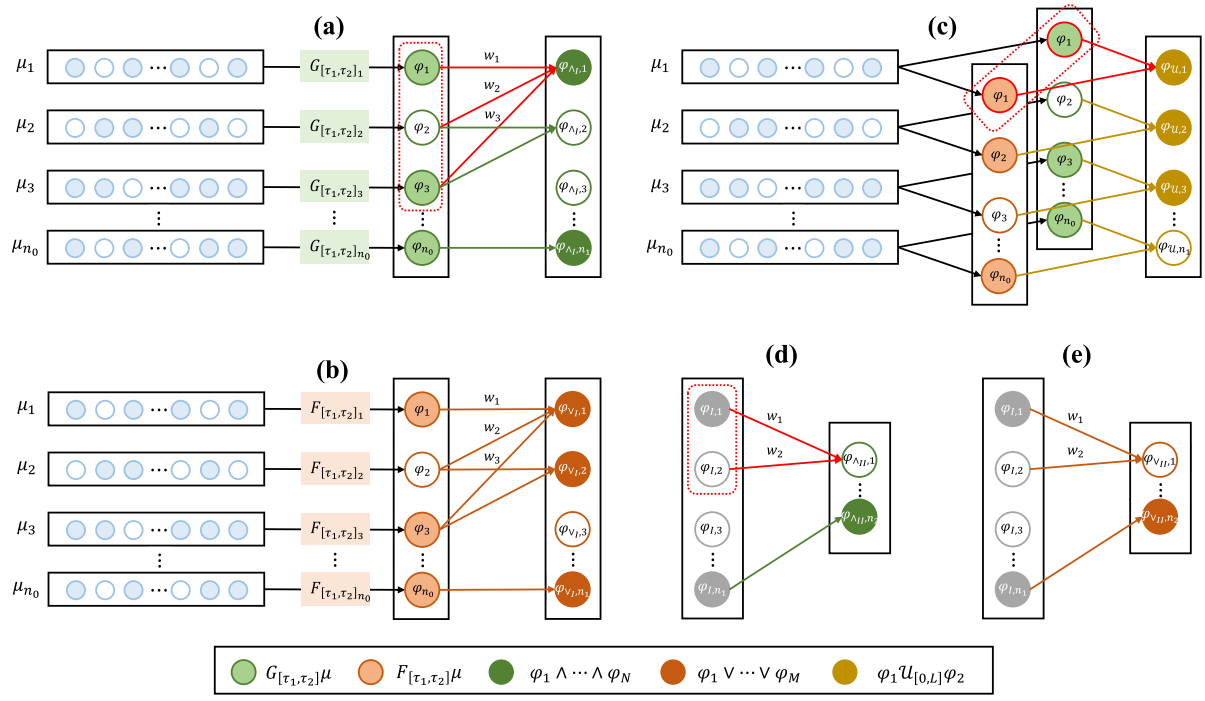


Fig. 4. Diagrams of temporal logic modules. (a) and (b) Temporal$_{A1}$ and Temporal$_{O1}$, which first add temporal operators before predicates, and then compose atomic formulas into Level I formulas. For example, in (a), three adjacent atomic formulas together form a Level I formula: $\varphi^{w_1}_{\wedge_I,1} = \varphi^{w_1}_1 \wedge \varphi^{w_2}_2 \wedge \varphi^{w_3}_3$. (c)–(e) Temporal$_{\mathcal{U}}$, Temporal$_{A2}$, and Temporal$_{O2}$, respectively, which use $\wedge$, $\vee$ or $\mathcal{U}$ to join Level I formulas into Level II formulas. In (d), two adjacent Level I formulas together form a Level II formula: $\varphi^{w_1}_{\wedge_{II},1} = \varphi^{w_1}_{I,1} \wedge \varphi^{w_2}_{I,2}$. The solid nodes represent valid wSTL formulas with robustness degrees greater than 0, the hollow nodes are invalid wSTL formulas with robustness degrees equal to 0.

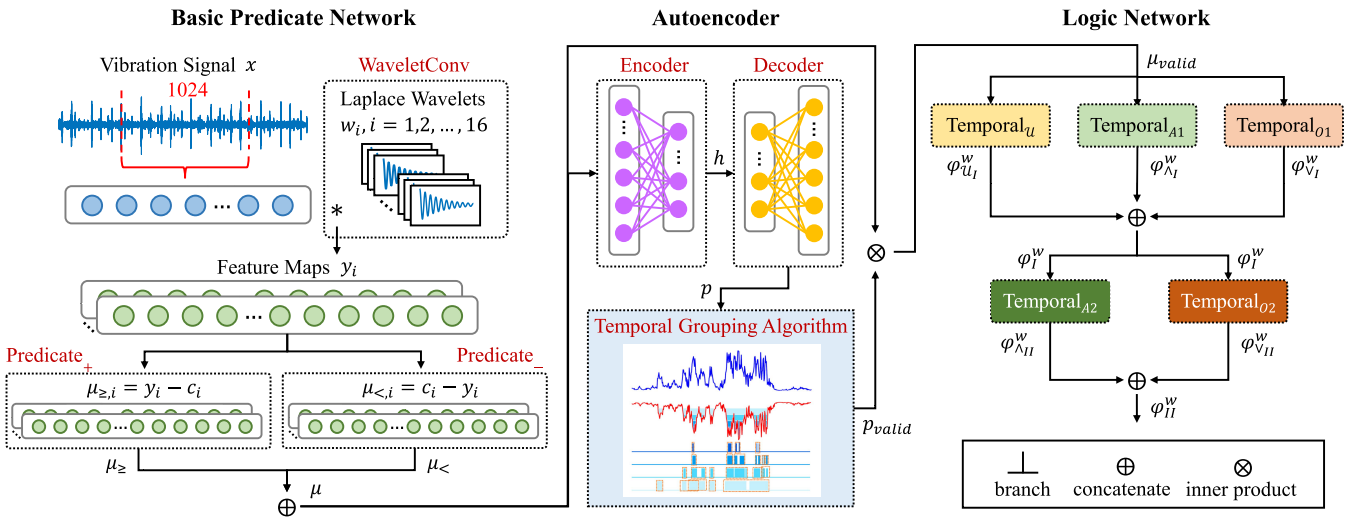


Fig. 5. Architecture of TLN.

where x denotes the vibration signal; and y denotes the convoluted time series. $\mu_{\geq}$ and $\mu_{<}$ represent the robustness degree of predicates with the form $y \geq c$ and $y < c$, respectively.

The autoencoder accepts the robustness degree sequence and calculates the probability that each sampling point falls within the effective time interval. The encoder is responsible for extracting abstract information from the robustness degree sequence. It has two compositions, each of which includes a convolutional layer, an activation layer, and a max-pooling layer. The decoder is responsible for mapping the abstract information to a probability sequence. It is similar to the encoder but processes the data in reverse order. Data are calculated in the autoencoder as follows:

$$h = \text{Encoder}(\mu), \quad p = \text{Decoder}(h) \tag{16}$$

where h is the hidden layer representation, and p denotes the probability sequence.

Section III states that a temporal logic operator typically has an interval parameter, representing the time span during which the operator takes effect. We use a temporal grouping algorithm for merging effective time intervals from the

probability sequence p, which is based on the watershed algorithm in [25].

As shown in Fig. 5 (temporal grouping algorithm modular), the blue curve is the probability sequence, and the red curve is its complement, on which the algorithm acts. The sequence is referred to as a "terrain" with "peaks" and "valleys." When water is poured into the terrain, a "water level" (γ) is created, and the flooded area is denoted by $G(\gamma)$. As the water level rises, adjacent flooded areas will become connected, which can be prevented by constructing a "dam" between them. The constructed dams can divide the terrain into several areas. In this analogy, "pouring water" and searching for the "flooded areas" refer to counting the effective intervals in the complementary sequence where the probability is less than γ; and constructing "dams" refers to merging the effective intervals. Note that the "valleys" in the red curve are the "peaks" in the blue curve, so the areas with lower water levels represent intervals with higher probability. In addition, we set the threshold v to represent the maximum proportion of the flooded area $G(\gamma)$ to the total area. When the proportion is larger than v, we create a watershed.

Using this algorithm, continuous and effective time intervals can be extracted from p

$$\mu_{\text{valid}} = p_{\text{valid}} \bigotimes \mu, \quad p_{\text{valid}} = \text{TG}(p) \tag{17}$$

where p_{valid} denotes the valid time intervals; μ_{valid} denotes the new robustness degree sequence; and $\bigotimes$ stands for the elemental product operation. In p_{valid}, if an element is 0, it means that the position is not in the effective interval; conversely, it is in the effective interval. Besides, the value of the nonzero element indicates the predicate's robustness degree.

In the logic network, we first use Temporal_{A1}, Temporal_{O1}, and $\text{Temporal}_{\mathcal{U}}$ to generate atomic formulas and connect them. Then we use Temporal_{A2} and Temporal_{O2} to combine the Level I formulas. Finally, wSTL formulas can be fed into a classifier and mapped to a certain bearing state. The computational steps of the logic network are represented as follows:

$$\begin{aligned}
&\varphi^w_{\wedge_I} = \text{Temporal}_{A1}(\mu_{\text{valid}}), \quad \varphi^w_{\vee_I} = \text{Temporal}_{O1}(\mu_{\text{valid}}) \\
&\varphi^w_{\mathcal{U}_I} = \text{Temporal}_{U}(\mu_{\text{valid}}), \quad \varphi^w_I = \text{concat}\left(\varphi^w_{\mathcal{U}_I}, \varphi^w_{\wedge_I}, \varphi^w_{\vee_I}\right) \\
&\varphi^w_{\wedge_{II}} = \text{Temporal}_{A2}\left(\varphi^w_I\right), \quad \varphi^w_{\vee_{II}} = \text{Temporal}_{O2}\left(\varphi^w_I\right) \\
&\varphi^w_{II} = \text{concat}\left(\varphi^w_{\wedge_{II}}, \varphi^w_{\vee_{II}}\right), \quad c = \text{Classifier}\left(\varphi^w_{II}\right)
\end{aligned} \tag{18}$$

where $\varphi^w_{\wedge_I}$, $\varphi^w_{\vee_I}$, and $\varphi^w_{\mathcal{U}_I}$ denote the Level I formulas connected by $\wedge$, $\vee$, and $\mathcal{U}$, respectively. $\varphi^w_{\wedge_{II}}$ and $\varphi^w_{\vee_{II}}$ denote the Level II formulas. c is the bearing state. The detailed configurations of the modules are shown in Table I.

V. Case Studies

In this section, three datasets are used to validate the effectiveness of the proposed interpretable model. What is more, two public datasets are used to perform comparative analysis of different methods. The signals are preprocessed in MATLAB environments, and all the results are obtained on an Intel[1] Core[2] i5-9300 CPU with 8.0-GB RAM via PyTorch, Python 3.8.

TABLE I
Configurations of the Modules

Module	Layer Components	Input Size	Output Size
WaveletConv	WaveConv(16,32)+ BN+ReLU+MaxPooling	$(1, 1024)$	$(16, 512)$
Predicate_+	Miu(16)+ MaxPooling	$(16, 512)$	$(16, 256)$
Predicate_-	Miu(16)+ MaxPooling	$(16, 512)$	$(16, 256)$
Encoder	Conv1d(32,64)+ Tanh+MaxPooling	$(32, 256)$	–
	Conv1d(64,32)+ ReLU+MaxPooling	–	–
Decoder	Upsample+ Conv1d(32,64)+Tanh	–	–
	Upsample+ Conv1d(64,32)+ Sigmoid+MaxPooling	–	$(32, 256)$
TG	Temporal Grouping	$(32, 256)$	$(160, 128)$
$\text{Temporal}_{\mathcal{U}}$	Until+ReLU	$(160, 128)$	(80)
$\text{Temporal}_{\text{A1}}$	And2d(20,10)+ReLU	$(160, 128)$	(45)
$\text{Temporal}_{\text{O1}}$	Or2d(20,10)+ReLU	$(160, 128)$	(45)
$\text{Temporal}_{\text{A2}}$	And1d(6,3)+ReLU	(170)	(45)
$\text{Temporal}_{\text{O2}}$	Or1d(6,3)+ReLU	(170)	(45)

A. CWRU Dataset

1) Dataset Introduction: The CWRU dataset was provided by Case Western Reserve University Bearing Data Center. In this dataset, electrical discharge machining (EDM) was used to create indentations on the bearing. The indentations are 0.007, 0.014, and 0.021 in in size, and they are positioned on the bearing's inner ring, outer ring, and rolling element, respectively. The rotating speeds of the bearing are 1797, 17726, 1750, and 1730 r/min. The motor loads are 0, 1, 2, and3 hp in each case. The sensor collected data at a sampling frequency of 12–48 kHz. In this article, we only use the vibration signals acquired at 0 hp with 1797 r/min and 12 kHz. More detailed descriptions of this dataset are presented in Table II.

2) Results: We intercepted segments from raw signals using a sliding window of length 1024. If the window length is too long, it extends the sequence involved in convolution, leading to longer operation times. Conversely, a too short window length results in incomplete information within a segment, limiting the model's ability to capture enough features or their relationships. For each bearing state, 80% of the samples are allocated for the training set and 20% for the testing set. We used Adam optimizer to update the model's parameters

[1]Registered trademark.
[2]Trademarked.

TABLE II
INFORMATION OF THE CWRU DATASET

Type	Symbol	Label	Size (in)	# Training	# Testing
Normal	N	0	0	191	48
Inner	IF_1	1	0.007	95	24
	IF_2	2	0.014	95	24
	IF_3	3	0.021	96	24
Outer	OF_1	4	0.007	96	24
	OF_2	5	0.014	95	24
	OF_3	6	0.021	96	24
Roller	RF_1	7	0.007	96	24
	RF_2	8	0.014	95	24
	RF_3	9	0.021	96	24

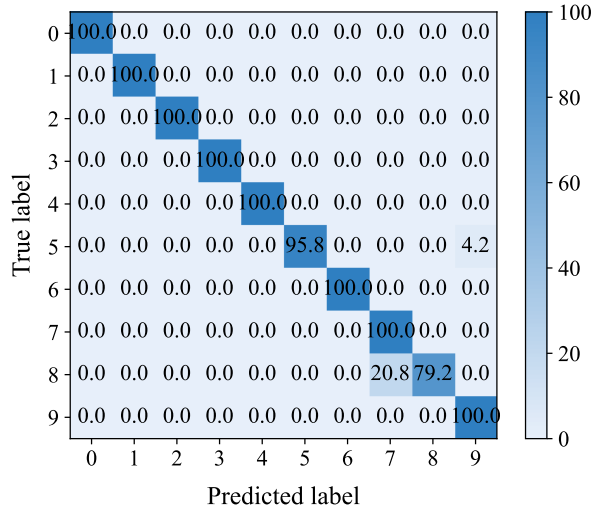


Fig. 6. Confusion matrix of the optimal model on the testing set. Labels 0-9 denote the ten states of the bearing.

with a learning rate of 0.0005. The network was trained for 400 epochs with a batch size of 20.

After training, the test accuracy of the model is 97.73%, indicating that TLN can not only distinguish fault modes but also has good generalization capability. Fig. 6 shows the confusion matrix of the optimal model, and the majority of faults can be accurately classified.

Fig. 7 illustrates the feature maps of TLN. The figures are organized into four rows, each representing a bearing state: normal, inner ring fault, outer ring fault, and rolling element fault. There are two figures for each state. The first one displays the vibration signal, its feature map, and the probability indicating where the features are located within the effective time intervals. First, the feature map inherits the peaks in the vibration signal, which means that TLN successfully extracts key features from the input. Second, based on real-life experience, different vibration signals will show different peaks at different times. As a result, the peak becomes one of the features used to describe the behavior of the signal. The feature map reveals a high probability that the peak is within the effective time intervals, which implies that TLN can position the intervals where the features are located. The second figure depicts the features' robustness degrees. The peaks have higher robustness degrees, which is compatible with the definition of wSTL's semantics.

Table III shows the wSTL formulas learned by TLN. The weights of the formulas are not displayed because they are not useful for interpretation. Moreover, we add a temporal operator $F_{[0,0.0010]}$ before each formula, representing that the formula will be satisfied within the first 0.001 s after the beginning of the signal. According to the definitions presented in Section II, we can explain each wSTL formula with a natural language sentence.

a) Normal: Within 0.001 s after the beginning of the signal, fault event φ_1 is expected to occur. φ_1 stands for the second dimension of $f(x)$, denoted as $f(x)_2$, which is always less than 0.9663 within the time interval of 0–0.032 s.

b) Inner ring fault: Within 0.001 s after the beginning of the signal, fault event φ_1 is expected to occur. Within 0.0063 s after the occurrence of φ_1, fault event φ_2 will occur. φ_1 stands for $f(x)_3$ is eventually larger than or equal to 0.1289 within 0.0078–0.0178 s. φ_2 denotes that $f(x)_{13}$ is eventually larger than or equal to 0.8452 within 0.0206–0.0226 s.

c) Outer ring fault: Within 0.001 s after the beginning of the signal, fault events φ_1 and φ_3 are expected to occur. Within 0.0063 s after the occurrence of φ_1, φ_2 will occur. Within 0.0063 s after the occurrence of φ_3, φ_4 will occur. φ_1 stands for $f(x)_7$ which is eventually larger than or equal to 0.3873 within 0–0.0010 s. φ_2 denotes that $f(x)_6$ is always larger than or equal to 0.0128 within 0.0048–0.0054 s. φ_3 stands for $f(x)_{16}$ is always larger than or equal to 0.0010 within 0.0215–0.0225 s. φ_4 denotes that $f(x)_{11}$ is eventually larger than or equal to 0.3886 within 0.0240–0.0254 s.

d) Rolling element fault: Within 0.001 s after the beginning of the signal, fault event φ_1 is expected to occur. Within 0.0063 s after the occurrence of φ_1, fault event φ_2 will occur. φ_1 stands for $f(x)_5$ is eventually larger than or equal to 0.0068 within 0.0076–0.0086 s. φ_2 denotes that $f(x)_7$ is eventually larger than or equal to 0.3873 within 0.0140–0.0171 s.

Fig. 8 displays the feature maps and TFPGs for the three fault states. As observed in the feature maps, the peaks in the vibration signals correspond to fault events, although not all the peaks indicate faults. This discrepancy may arise due to the sufficient discriminative power of a smaller number of features when performing fault classification. The TLN, recognizing the low average energy and absence of prominent peaks in the normal state signal, learns a simpler wSTL formula that only requires the signal's amplitude not to exceed a certain constant threshold. Moreover, the TFPGs vividly depict the logical connections between fault events and their temporal diffusion. In contrast to alternative visualization techniques, the proposed approach affords a more uncomplicated and accessible representation, enabling personnel without specialized expertise to comprehend the practical ramifications of the failures. The rigorous capture of signal impulses by the wSTL formula empowers workers to detect faults accurately and make informed decisions regarding part replacement or lubrication. However, due to the long runtime of the temporal logic modules and several convolutional operations involved, the training time of TLN is long.

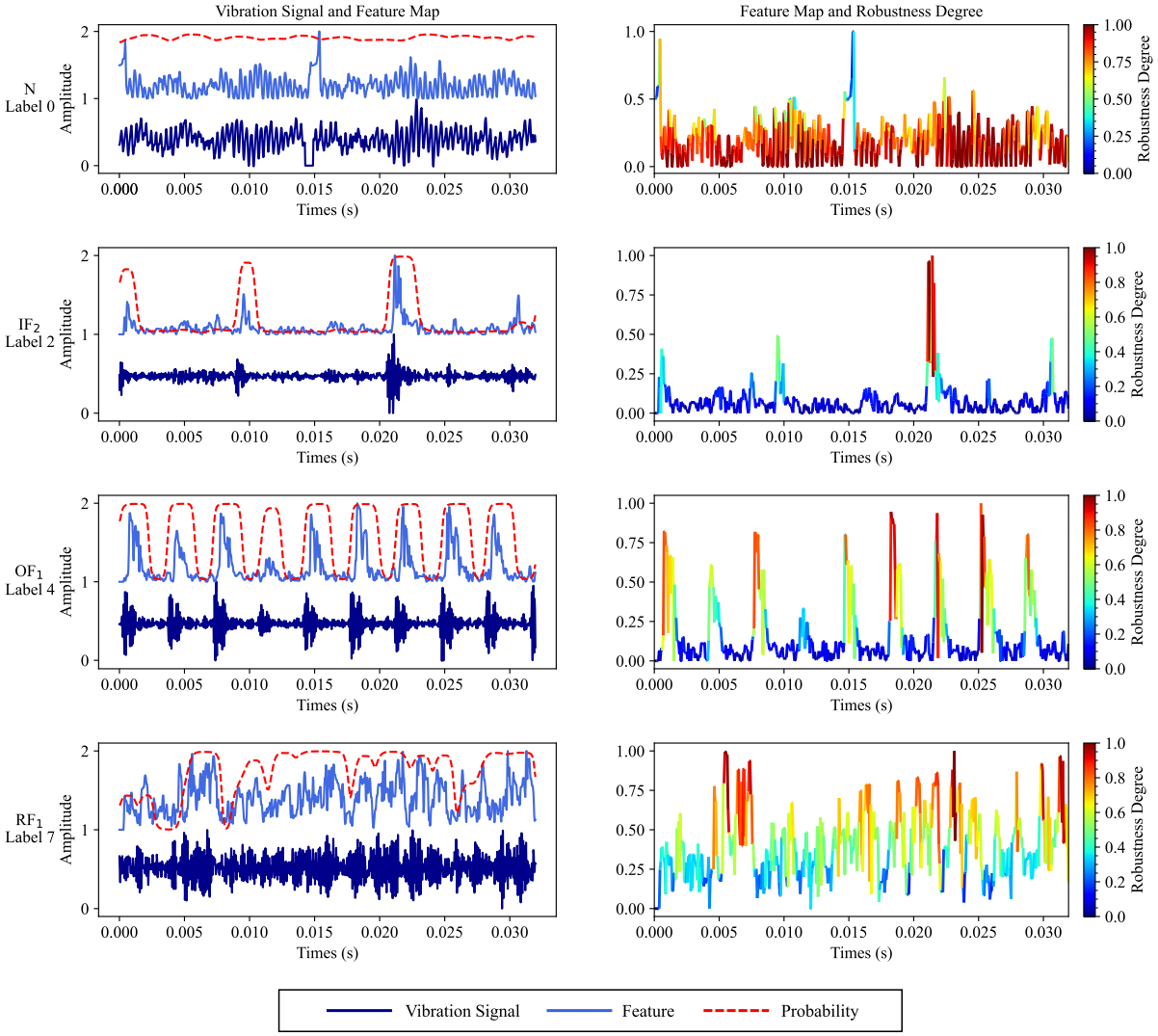


Fig. 7. TLN's feature maps after trained with the CWRU dataset. Both features and robustness degrees are normalized.

TABLE III
wSTL FORMULAS FOR FOUR BEARING STATES IN THE CWRU DATASET

Bearing State	wSTL formula
Normal	$\varphi_N = F_{[0,0.0010]}(G_{[0,0.0320]} f(x)_2 < 0.9663)$
Inner Ring Fault	$\varphi_{IF_2} = F_{[0,0010]}(\varphi_1 \mathcal{U}_{[0,0.0063]} \varphi_2)$, where $\varphi_1 = F_{[0.0078,0.0178]}(f(x)_3 \geq 0.1289)$, $\varphi_2 = F_{[0.0206,0.0226]}(f(x)_{13} \geq 0.8452)$
Outer Ring Fault	$\varphi_{OF_1} = F_{[0,0010]}((\varphi_1 \mathcal{U}_{[0,0.0063]} \varphi_2) \wedge (\varphi_3 \mathcal{U}_{[0,0.0063]} \varphi_4))$, where $\varphi_1 = F_{[0,0.0010]}(f(x)_7 \geq 0.3873)$, $\varphi_2 = G_{[0.0048,0.0054]}(f(x)_6 \geq 0.0128)$, $\varphi_3 = G_{[0.0215,0.0225]}(f(x)_{16} \geq 0.0010)$, $\varphi_4 = F_{[0.0240,0.0254]}(f(x)_{11} \geq 0.3886)$
Rolling Element Fault	$\varphi_{RF_1} = F_{[0,0010]}(\varphi_1 \mathcal{U}_{[0,0.0063]} \varphi_2)$, where $\varphi_1 = F_{[0.0076,0.0086]}(f(x)_5 \geq 0.0068)$, $\varphi_2 = F_{[0.0140,0.0171]}(f(x)_7 \geq 0.3873)$

B. SCUT Dataset

1) Dataset Introduction: CWRU is a high-quality dataset with small noise. To assess the generalizability and robustness of TLN, we evaluated the model's performance using a dataset that we have collected. The SCUT dataset was provided by South China University of Technology, and the test rig is shown in Fig. 9. It has a motor, an electromagnetic brake that can apply a torque load, and a hydraulic jack that can apply a radial load. The indentation size is 0.8 mm, and they are positioned on the bearing's inner ring, outer ring, and rolling element, respectively (as shown in Fig. 10). The vibration signals are sampled at the bearing fixture by two mutually perpendicular accelerometers in the horizontal and vertical directions under four speeds (1240, 1210, 1180, and 1150 r/min) with a sampling frequency of 32 kHz. The bearing state can be divided into four classes, and more detailed descriptions of this dataset are presented in Table IV.

2) Results: Initially, TLN was trained using time-domain signals as the training data, keeping the experimental configurations unchanged. However, the classification accuracy was relatively low, which is only around 80%. The reasons

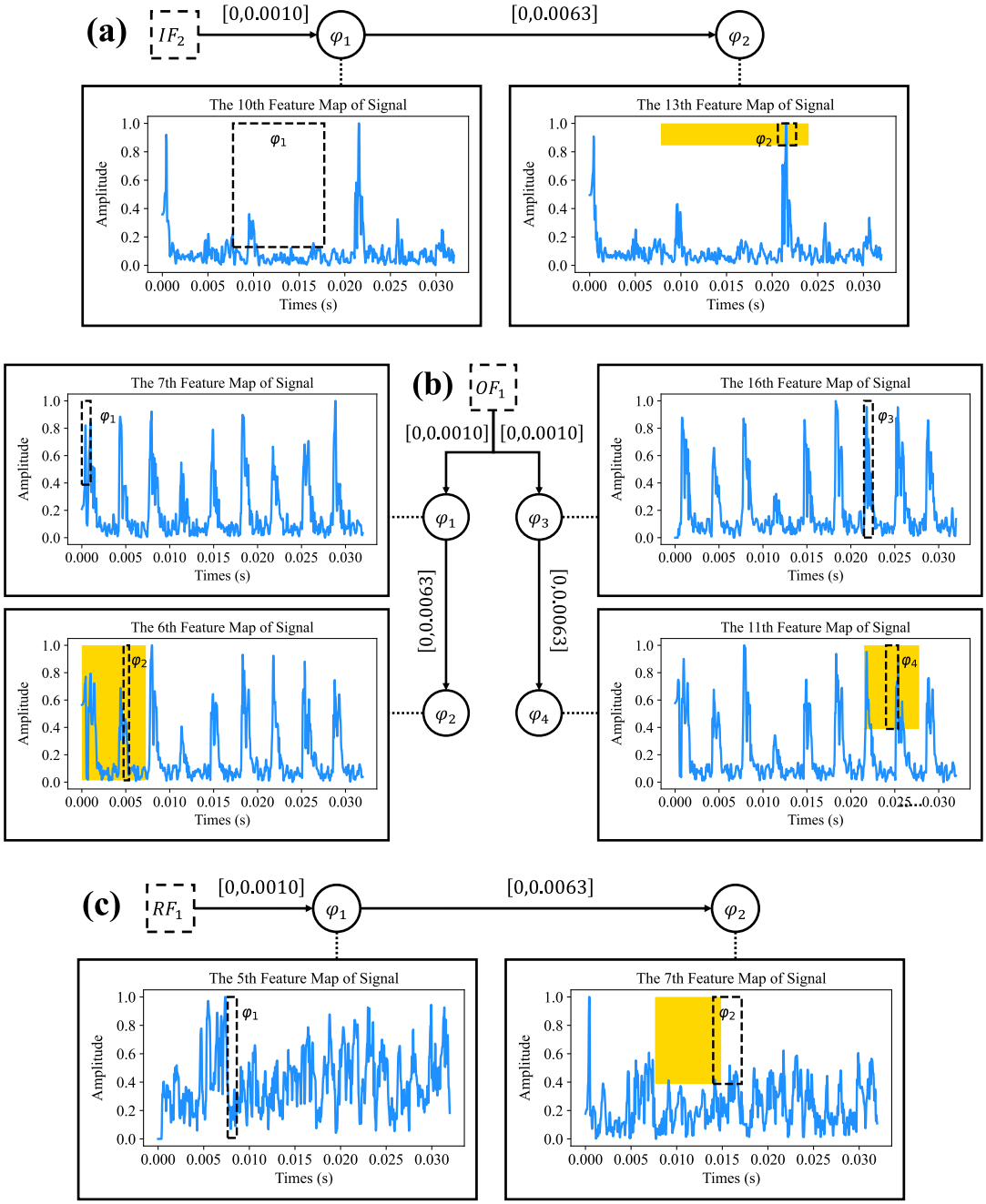


Fig. 8. TFPGs mapped by wSTL formulas. (a) TFPG of inner ring fault. (b) TFPG of outer ring fault. (c) TFPG of rolling element fault. Each TFPG not only has a basic frame but also has a feature map attached to the discrepancy node. The black dashed box frames the absolute intervals of the fault event, and the yellow area represents the interval in which the event may occur. For the inner ring fault, there are two fault events. φ_1 means that $f(x)_3$ will eventually be greater than or equal to 0.1289 in 0.0078–0.0178 s, whereas φ_2 means that $f(x)_{13}$ will eventually be greater than or equal to 0.8452 in 0.0206–0.0226 s. According to the definition of wSTL, the theoretical existence interval of φ_2 is 0.0078–0.0240, as covered by the yellow area. The left line of the black dashed box lies within the yellow region, indicating that the wSTL formula generated by the TLN is grammatically correct.

could be: First, the SCUT dataset, as opposed to the CWRU dataset, contains a higher degree of noise, leading to less prominent distinctions among the waveforms of different signal types, as shown in Fig. 11. Second, the noise in vibration signals is not stationary, resulting in varying peak positions even among samples of the same type, thereby diminishing their commonalities.

To address these challenges, we devised an alternative approach wherein TLN was trained using frequency-domain data. After applying a low-pass filter for noise reduction, we first split the filtered signal into segments of length 1024. Then we computed their power spectra within the frequency range of 0–4000 Hz. Finally, the normalized power spectra were then used as the training data. In this experiment, we omitted the wavelet convolution module for feature extraction, given that the input data had already been filtered. Furthermore, since the training data were in the frequency domain, we excluded the temporal logic module $\text{Temporal}_{\mathcal{U}}$

TABLE IV
INFORMATION OF THE SCUT DATASET

Bearing State	Symbol	Label	Speed (rpm)	Indentation Size (mm)	# Training/Testing Samples	$\text{Accuracy}_{\text{time}}$ (Training/Testing, %)	$\text{Accuracy}_{\text{freq}}$ (Training/Testing, %)
Normal	N	0	1240	0.8	641/160	92.55/87.50	99.38/100.00
Inner ring Fault	IF	1	1210	0.8	641/160	87.58/85.00	98.28/100.00
Outer ring Fault	OF	2	1180	0.8	641/160	90.06/97.50	99.69/100.00
Rolling element Fault	RF	3	1150	0.8	641/160	56.52/72.50	100.00/100.00

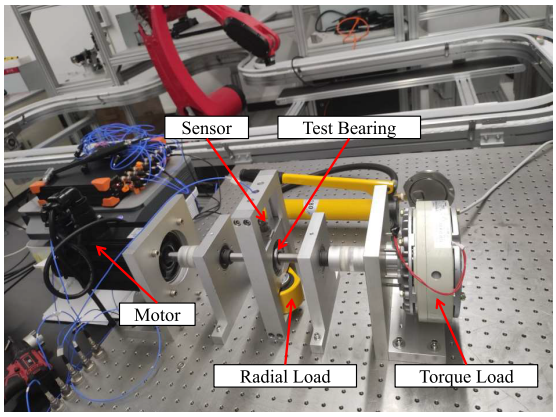


Fig. 9. Bearing test rig of the SCUT dataset.

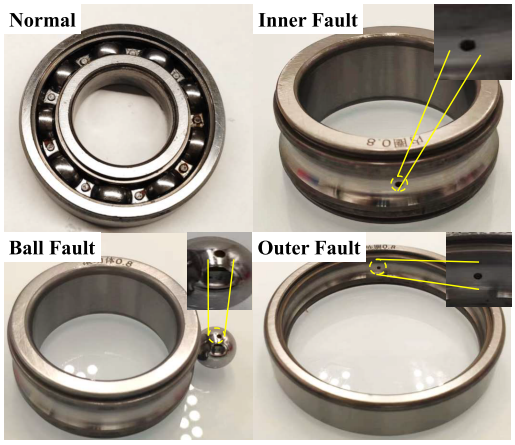


Fig. 10. Four states of the rolling element bearing.

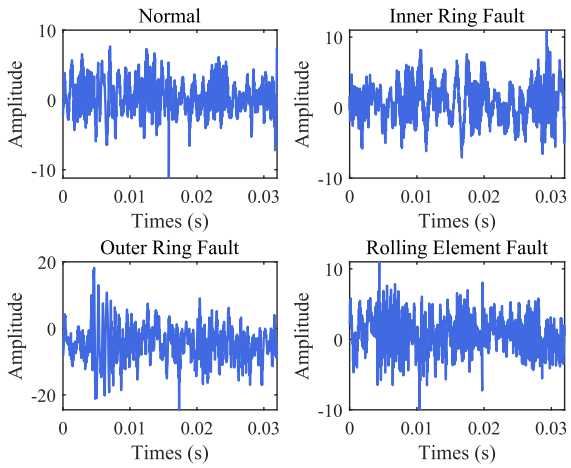


Fig. 11. Waveforms of the signals within the SCUT dataset.

from the TLN architecture. We opted for the Adam optimizer to update the model's parameters, setting the learning rate at 0.0001. The network underwent training for 400 epochs with a batch size of 20. Across all the bearing states, the model trained using frequency-domain data consistently outperforms its time-domain counterpart, as depicted in Table IV. The former boasts an average accuracy of 100.00%, whereas the latter demonstrates an average accuracy of only 85.62%.

Fig. 12 presents the normalized feature maps of the signals. It is observed that frequency bands with higher power densities are more likely to be signal features, indicating that TLN can identify signal characteristic frequencies. Moreover, the higher robustness degrees of the power spectra peaks demonstrate the effectiveness of the formal formula in describing the signal's behavior, particularly its peaks' magnitude. The formulas learned by TLN are displayed in Table V, and the subscripts of the temporal operator signify the frequency interval where the predicate applies. Below are the interpretations of these formulas in natural language.

a) Normal: The behavior of the signal has to satisfy the description of the atomic formula φ_1, which means that the power density is always greater than or equal to 0.4028 W/Hz within the frequency interval of 732.27–841.32 Hz.

b) Inner ring fault: The behavior of the signal has to satisfy at least one of the descriptions of these three atomic formulas: φ_1, φ_2, and φ_3. φ_1 means that the power density is eventually larger than or equal to 0.3266 W/Hz within 15.63–77.94 Hz. φ_2 means that the power density is eventually larger than or equal to 0.0006 W/Hz within 794.58–903.64 Hz. φ_3 means that the power density is eventually larger than or equal to 0.0006 W/Hz within 2118.81–2227.87 Hz.

c) Outer ring fault: The power density of the signal is always greater than or equal to 0.3415 W/Hz within 1667.02–1744.91 Hz.

d) Rolling element fault: The behavior of the signal has to satisfy at least one of the descriptions of these three atomic formulas: φ_1, φ_2, and φ_3. φ_1 means that the power density is eventually larger than or equal to 0.3415 W/Hz within 15.63–109.10 Hz. φ_2 means that the power density is eventually larger than or equal to 0.4134 W/Hz within 716.69–825.74 Hz. φ_3 means that the power density is eventually larger than or equal to 0.0539 W/Hz within 1776.07–1853.97 Hz.

TABLE V
wSTL FORMULAS FOR FOUR BEARING STATES IN THE SCUT DATASET

Bearing State	wSTL formula
Normal	$\varphi_N = G_{[732.27,841.32]}(f(x) \geq 0.4028)$
Inner Ring Fault	$\varphi_{IF} = \varphi_1 \vee \varphi_2 \vee \varphi_3$ $\varphi_1 = F_{[15.63,77.94]}(f(x) \geq 0.3266)$, $\varphi_2 = F_{[794.58,903.64]}(f(x) \geq 0.0006)$, $\varphi_3 = F_{[2118.81,2227.87]}(f(x) \geq 0.0006)$
Outer Ring Fault	$\varphi_{OF} = G_{[1667.02,1744.91]}(f(x) \geq 0.3415)$
Rolling Element Fault	$\varphi_{RF} = \varphi_1 \vee \varphi_2 \vee \varphi_3$ $\varphi_1 = F_{[15.63,109.10]}(f(x) \geq 0.3415)$, $\varphi_2 = F_{[716.69,825.74]}(f(x) \geq 0.4134)$, $\varphi_3 = F_{[1776.07,1853.97]}(f(x) \geq 0.0539)$

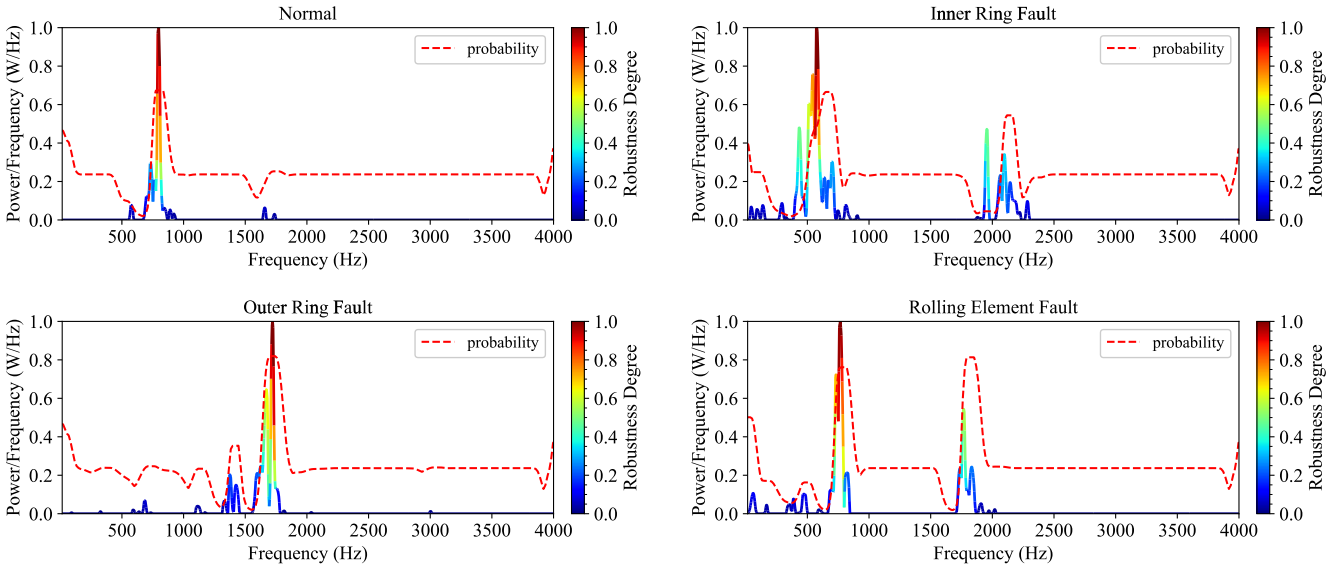


Fig. 12. TLN's feature maps after trained with the SCUT dataset. In this experiment, the feature maps are the power spectra of the vibration signals.

The power spectra for the four bearing states are presented in Fig. 13. Since frequency-domain data are s used in this experiment, TFPGs are no longer used to show the propagation of the fault events. What the wSTL formulas describe are the frequency bands in which the power is concentrated, indicating that the proposed model captures the signal's characteristic frequencies.

C. Comparisons With Other Methods

To verify the adaptability of TLN, we chose another public dataset, the machinery failure prevention technology (MFPT) dataset, to test the model. This dataset is composed of four sets of bearing vibration signals.

1) *Baseline Conditions:* The data are collected at 270 lbs of load and a sampling rate of 97 656 Hz for 6 s.
2) *Outer Race Fault:* The data are collected at 270 lbs of load and a sampling rate of 97 656 Hz for 6 s.
3) *Outer Race Fault:* The data are collected at 7 different loads and a sampling rate of 48 828 Hz for 3 s.
4) *Inner Race Fault:* The data are collected at 7 different loads and a sampling rate of 48 828 Hz for 3 s.

The shafts all rotated at 25 Hz. Therefore, the signals within this dataset can be categorized into three states: normal, inner ring fault, and outer ring fault.

TABLE VI
COMPARATIVE RESULTS OF THE CWRU AND MFPT DATASETS

Model	Method	CWRU (%)	MFPT (%)
ANN[26]	ZC	91.50 ∼ 97.10	–
DCNN[27]	STFT	–	99.90
	WT	–	91.30
	HHT	–	92.90
CNN[28]	STMSST	99.83	98.67
SNN[29]	LMD	99.17	99.54
CNN+PCA+FCM[13]	CNN+PCA	99.30 ∼ 100.00	–
CISTA-Net[30]	-	100.00	–
ResCISTA-Net[30]	-	–	95.43
TLN	WT+wSTL	97.73	98.86

We selected normal data, inner ring fault data with a load of 50 lbs, and outer ring fault data with a load of 300 lbs from the MFPT dataset for model training and testing. Data preprocessing and model configuration are the same in Section V-A. After obtaining the test accuracy, we compare the experimental results of TLN on the CWRU and MFPT datasets with other methods, which are presented in Table VI.

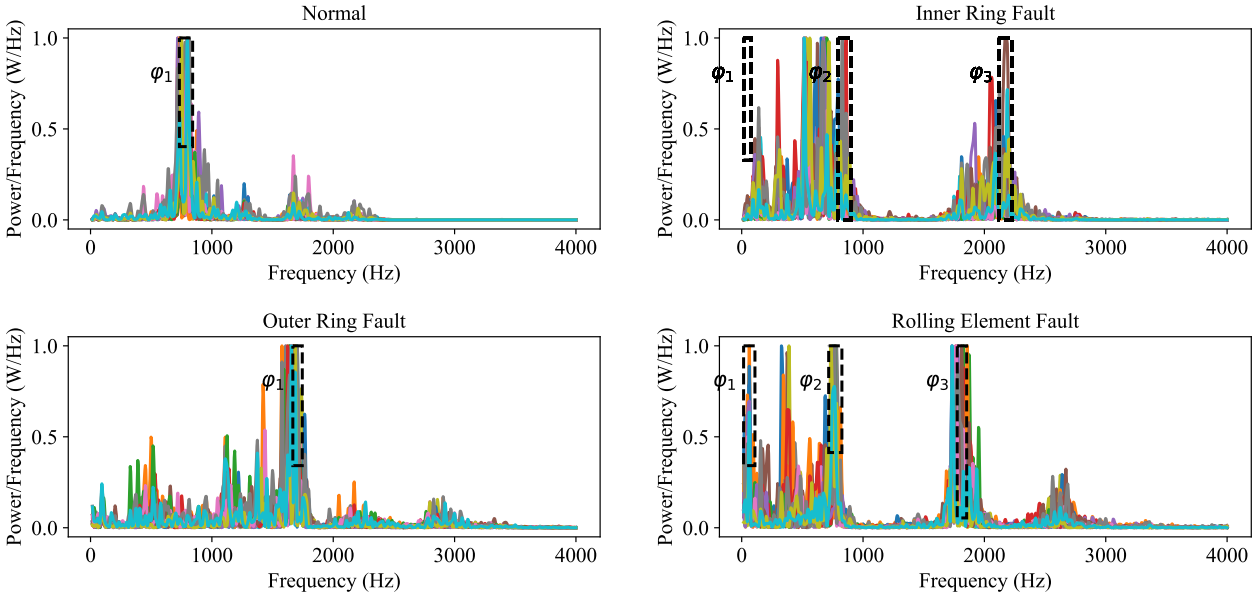


Fig. 13. Signal power spectra in four states. Black dashed blocks frame the fault events, and 20 power spectra are plotted for each bearing state.

In [26], zero-crossing (ZC) is used for extracting features from raw signals, followed by classification using a feedforward artificial neural network (ANN). Verstraete et al. [27] use short-time Fourier transform (STFT), wavelet transform (WT), and Hilbert–Huang transform (HHT) to generate time–frequency images of the signals, and then feed them into a deep convolutional CNN (DCNN) for fault diagnosis. In [28], a new approach named second-order time-reassigned multisynchrosqueezing transform (STMSST) based on Gaussian-modulated linear group delay (GLGD) converts 1-D signals into 2-D images for feature extraction, and these time–frequency images are input into a CNN for classification. Zuo et al. [29] use local mean decomposition (LMD) to extract features, encoding them into spikes for classification through a spiking neural network (SNN). The test accuracy of TLN is 98.86%, which is above ANN, DCNN with WT, and HHT. Although the proposed model's accuracy is 2.27% and 1.04% lower than the best-performing models on the CWRU and MFPT datasets, it is interpretable, which is a characteristic that the aforementioned models do not have. Zhang et al. [13] use CNN and principal component analysis (PCA) to extract feature vectors from vibration signals and then cluster them using the fuzzy C-means (FCMs) algorithm. Rao et al. [30] use the algorithm unrolling technique to unroll the convolutional version of the iterative shrinkage-thresholding algorithm (CISTA) into a neural network called CISTA-Net, and then add a residual block at the network's input to create ResCISTA-Net. It can be seen that the accuracy of TLN is higher than that of ResCISTA-Net. Both the methods proposed in [13] and [30] can extract interpretable features, whereas TLN can also identify interpretable logical relationships and propagation between the features, allowing fault events to be intuitively understood.

D. Discussion

TLN's primary advantage lies in its ability to elucidate the temporal dynamic properties inherent in time-series data. This is particularly crucial in bearing fault diagnosis, where understanding how faults evolve over time is key to effective prediction and prevention. TLN's approach to interpreting time-series data goes beyond static analysis, offering deeper insights into the sequential and temporal patterns that the traditional models might overlook. While TLN offers advanced interpretability in temporal dynamics, it currently does not match the accuracy and efficiency of other deep learning models. In the context of bearing fault diagnosis, where quick and precise detection is essential, this presents a challenge. The intricate nature of TLN's temporal analysis, though insightful, may contribute to this discrepancy in performance. Integrating advanced signal-processing techniques could enhance TLN's accuracy. These techniques can help in better capturing the complexities of time-series data, potentially improving TLN's diagnostic capabilities. Future improvements for TLN should aim at refining its ability to analyze temporal dynamics more efficiently without compromising on interpretability. Optimizing the network's architecture and incorporating signal-processing methods are viable pathways. Exploring hybrid models that combine the temporal interpretability of TLN with the accuracy of conventional deep learning models could also be beneficial.

VI. Conclusion

In this article, we rigorously define a formal language named weighted temporal logic and propose a deep learning architecture called TLN for interpretable fault diagnosis of rolling element bearings. TLN can symbolize vibration signals and map them to a wSTL formula. To further validate the interpretability of the model, timed failure propagation graphs (TFPGs) are used to describe the logical relationship and propagation between fault events in the time domain. Experiments on three datasets and comparisons with other models show that the proposed method has good performance for fault diagnosis and interpretability. The experimental results also show that TLN can extract the impulse patterns of a signal in either the time or frequency domains, which is better than other methods in terms of accuracy and interpretability.

REFERENCES

[1] G. Yu, "A concentrated time–frequency analysis tool for bearing fault diagnosis," *IEEE Trans. Instrum. Meas.*, vol. 69, no. 2, pp. 371–381, Feb. 2020.

[2] J. Zhang, K. Zhang, Y. An, H. Luo, and S. Yin, "An integrated multitasking intelligent bearing fault diagnosis scheme based on representation learning under imbalanced sample condition," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, pp. 1–12, Jan. 2023.

[3] J. A. Reyes-Malanche, F. J. Villalobos-Pina, E. Cabal-Yepez, R. Alvarez-Salas, and C. Rodriguez-Donate, "Open-circuit fault diagnosis in power inverters through currents analysis in time domain," *IEEE Trans. Instrum. Meas.*, vol. 70, pp. 1–12, 2021.

[4] I. Attoui, N. Boutasseta, and N. Fergani, "Novel machinery monitoring strategy based on time–frequency domain similarity measurement with limited labeled data," *IEEE Trans. Instrum. Meas.*, vol. 70, pp. 1–8, 2021.

[5] K. Yue, J. Li, J. Chen, R. Huang, and W. Li, "Multiscale wavelet prototypical network for cross-component few-shot intelligent fault diagnosis," *IEEE Trans. Instrum. Meas.*, vol. 72, pp. 1–11, 2023.

[6] S. Zhang, S. Zhang, B. Wang, and T. G. Habetler, "Deep learning algorithms for bearing fault diagnostics—A comprehensive review," *IEEE Access*, vol. 8, pp. 29857–29881, 2020.

[7] J.-X. Liao, H.-C. Dong, Z.-Q. Sun, J. Sun, S. Zhang, and F.-L. Fan, "Attention-embedded quadratic network (Qttention) for effective and interpretable bearing fault diagnosis," *IEEE Trans. Instrum. Meas.*, vol. 72, pp. 1–13, 2023.

[8] Z. Zhao et al., "Model-driven deep unrolling: Towards interpretable deep learning against noise attacks for intelligent fault diagnosis," *ISA Trans.*, vol. 129, pp. 644–662, Oct. 2022.

[9] I. Kim et al., "Single domain generalizable and physically interpretable bearing fault diagnosis for unseen working conditions," *Expert Syst. Appl.*, vol. 241, May 2024, Art. no. 122455.

[10] J. Tang, G. Zheng, C. Wei, W. Huang, and X. Ding, "Signal-transformer: A robust and interpretable method for rotating machinery intelligent fault diagnosis under variable operating conditions," *IEEE Trans. Instrum. Meas.*, vol. 71, pp. 1–11, 2022.

[11] M. A. Shafiq, Z. Long, H. Di, G. A. Regib, and M. Deriche, "Fault detection using attention models based on visual saliency," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2018, pp. 1508–1512.

[12] M. S. Kim, J. P. Yun, and P. Park, "An explainable neural network for fault diagnosis with a frequency activation map," *IEEE Access*, vol. 9, pp. 98962–98972, 2021.

[13] D. Zhang, Y. Chen, F. Guo, H. R. Karimi, H. Dong, and Q. Xuan, "A new interpretable learning method for fault diagnosis of rolling bearings," *IEEE Trans. Instrum. Meas.*, vol. 70, pp. 1–10, 2021.

[14] K.-S. Kang, C. Koo, and H.-G. Ryu, "An interpretable machine learning approach for evaluating the feature importance affecting lost workdays at construction sites," *J. Building Eng.*, vol. 53, Aug. 2022, Art. no. 104534.

[15] M. Angelini, G. Blasilli, S. Lenti, and G. Santucci, "A visual analytics conceptual framework for explorable and steerable partial dependence analysis," *IEEE Trans. Vis. Comput. Graphics*, early access, pp. 1–16, Apr. 2023.

[16] E. Brusa, L. Cibrario, C. Delprete, and L. G. D. Maggio, "Explainable AI for machine fault diagnosis: Understanding features' contribution in machine learning models for industrial condition monitoring," *Appl. Sci.*, vol. 13, no. 4, p. 2038, Feb. 2023.

[17] A. Movsessian, D. G. Cava, and D. Tcherniak, "Interpretable machine learning in damage detection using Shapley additive explanations," *ASCE-ASME J. Risk Uncertainty Eng. Syst. Part B, Mech. Eng.*, vol. 8, no. 2, Jun. 2022, Art. no. 021101.

[18] X. Wang, H. Gu, T. Wang, W. Zhang, A. Li, and F. Chu, "Deep convolutional tree-inspired network: A decision-tree-structured neural network for hierarchical fault diagnosis of bearings," *Frontiers Mech. Eng.*, vol. 16, no. 4, pp. 814–828, Dec. 2021.

[19] F. Pohlmeyer, R. Kins, F. Cloppenburg, and T. Gries, "Interpretable failure risk assessment for continuous production processes based on association rule mining," *Adv. Ind. Manuf. Eng.*, vol. 5, Nov. 2022, Art. no. 100095.

[20] F. Li, J. Chen, S. He, and Z. Zhou, "Layer regeneration network with parameter transfer and knowledge distillation for intelligent fault diagnosis of bearing using class unbalanced sample," *IEEE Trans. Instrum. Meas.*, vol. 70, pp. 1–10, 2021.

[21] R. de Paula Monteiro, M. C. Lozada, D. R. C. Mendieta, R. V. S. Loja, and C. J. A. B. Filho, "A hybrid prototype selection-based deep learning approach for anomaly detection in industrial machines," *Expert Syst. Appl.*, vol. 204, Oct. 2022, Art. no. 117528.

[22] G. Chen, P. Wei, H. Jiang, and M. Liu, "Formal language generation for fault diagnosis with spectral logic via adversarial training," *IEEE Trans. Ind. Informat.*, vol. 18, no. 1, pp. 119–129, Jan. 2022.

[23] G. Chen, M. Liu, and Z. Kong, "Temporal-logic-based semantic fault diagnosis with time-series data from industrial Internet of Things," *IEEE Trans. Ind. Electron.*, vol. 68, no. 5, pp. 4393–4403, May 2021.

[24] G. Chen, M. Liu, and J. Chen, "Frequency-temporal-logic-based bearing fault diagnosis and fault interpretation using Bayesian optimization with Bayesian neural networks," *Mech. Syst. Signal Process.*, vol. 145, Nov. 2020, Art. no. 106951.

[25] Y. Zhao, Y. Xiong, L. Wang, Z. Wu, X. Tang, and D. Lin, "Temporal action detection with structured segment networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2914–2923.

[26] P. E. William and M. W. Hoffman, "Identification of bearing faults using time domain zero-crossings," *Mech. Syst. Signal Process.*, vol. 25, no. 8, pp. 3078–3088, Nov. 2011.

[27] D. Verstraete, A. Ferrada, E. L. Droguett, V. Meruane, and M. Modarres, "Deep learning enabled fault diagnosis using time-frequency image analysis of rolling element bearings," *Shock Vibrat.*, vol. 2017, pp. 1–17, Oct. 2017.

[28] G. Sun, Y. Gao, Y. Xu, and W. Feng, "Data-driven fault diagnosis method based on second-order time-reassigned multisynchrosqueezing transform and evenly mini-batch training," *IEEE Access*, vol. 8, pp. 120859–120869, 2020.

[29] L. Zuo, L. Zhang, Z.-H. Zhang, X.-L. Luo, and Y. Liu, "A spiking neural network-based approach to bearing fault diagnosis," *J. Manuf. Syst.*, vol. 61, pp. 714–724, Oct. 2021.

[30] F. Rao, M. Zeng, and Y. Cheng, "A novel interpretable model via algorithm unrolling for intelligent fault diagnosis of machinery," *IEEE Sensors J.*, vol. 24, no. 1, pp. 495–505, Nov. 2023.

Ruoyao Tian received the bachelor's degree in robotic engineering from the South China University of Technology, Guangzhou, China, in 2023. She is currently pursuing the master's degree in mechanical engineering with Zhejiang University, Hangzhou, China.

Her research interests include deep learning and fault diagnosis.

Mengqian Cui received the bachelor's degree in economic statistics from the Henan University of Economics and Law, Zhengzhou, China, in 2017, and the master's degree in applied statistics from Jinan University, Guangzhou, China, in 2019.

She is working with the Guangdong Rural Credit Union, Guangdong, China. Her research interests include machine learning, data security, and fault detection.

Gang Chen (Member, IEEE) received the bachelor's and master's degrees in mechanical engineering from Shanghai Jiao Tong University, Shanghai, China, in 2012 and 2015, respectively, and the Ph.D. degree in mechanical and aerospace engineering from the University of California at Davis, Davis, CA, USA, in 2020.

He was a Research Fellow with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore, from 2020 to 2021. He is currently an Associate Professor with the Shien-Ming Wu School of Intelligent Engineering, South China University of Technology, Guangzhou, China. His research interests include machine learning, formal methods, control, signal processing, and fault diagnosis.