IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 27 May 2022, accepted 7 June 2022, date of publication 21 June 2022, date of current version 28 June 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3184693

A Deep Explainable Model for Fault Prediction Using IoT Sensors

TAHA MANSOURI, (Member, IEEE), AND SUNIL VADERA
School of Science, Environment, and Engineering, University of Salford, Manchester M5 4WT, U.K.

Corresponding author: Taha Mansouri (t.mansouri@salford.ac.uk)

The data used in the case study arose as part of the Knowledge Transfer Partnership between the University of Salford and Invisible systems Ltd (ISL) partially funded by Innovate UK "KTP011129".

ABSTRACT IoT sensors and deep learning models can widely be applied for fault prediction. Although deep learning models are considerably more potent than many conventional machine learning models, they are not transparent. This paper first examines different deep learning techniques to carry out univariate time series analysis based on vibration sensors installed on four industrial bearings to predict a fault occurring in a predefined time window. Several recurrent neural networks are used to develop fault prediction models. An empirical evaluation of these models shows that all models perform well; however, hybrid models outperform other models when the time window increases. Then, instance-wise feature selection has been considered to highlight the most contributing features for its outputs regarding any input. In this problem, the main challenge is to propose a trainable feature selection model with the minimum number of selected features whilst its performance is close to the baseline model. This paper develops a novel explainable method called the Gumbel-Sigmoid eXplanator (GSX) to tackle these problems. In a nutshell: (i) we have developed a differentiable and trainable selector, and (ii) we utilize regularization to control the number of features for each instance flexibly. The proposed method is model agnostic, and empirical evaluations on two datasets show that GSX can not only solve the problems identified with two other state-of-the-art methods but also outperform them in terms of accuracy and run-time.

INDEX TERMS Deep learning, explainable AI, fault prediction, predictive preventive maintenance.

I. INTRODUCTION

Nowadays, in light of emerging technologies such as 5G, cloud computing, and fast-growing sensor manufacturing technologies, the Internet of Things is getting more popular. Among a wide variety of applications, condition monitoring is a demanding IoT application [1], [2]. In this use case, different sensors are installed on equipment to monitor various aspects to identify their condition or conduct analytical methods to predict a fault in the future [3], [4]. The vibration signals are measured almost in all equipment in power plants; therefore, they are available in most systems [5]. Moreover, measuring vibration signals can lead to retrieving valuable insight into the condition of the equipment and fault prediction by recognizing the underlying patterns [4]. Therefore, vibration signals are common sources for monitoring equipment conditions [4]–[11]. There are two major approaches to fault prediction, including physics-based and data-driven models [12], [13]. Physics-based models mostly use the knowledge of domain experts and the logic behind measured data to construct a mathematical model [12]. This approach suffers from two main problems; first, its performance depends on the accuracy of the available domain knowledge, and second, it cannot be promptly updated [13]. Contrary, data-driven models concentrate on extracting underlying patterns among the online data fetched from sensors by machine learning and analytical techniques. Therefore, they can automatically extract the knowledge from data, and their parameters can be updated instantly when the value of incoming data streams changes [12], [13]. So, among various techniques to carry out predictive preventive maintenance, data-driven approaches, including statistical and machine learning methods, are gaining popularity [12]. In conventional machine learning, feature selection is the primary challenge. To tackle this problem, most reported works have combined a wavelet transformation with other techniques such as self-organizing maps [14], neural networks [15], [16], and SVMs [5]. Evolutionary computing also has been deployed for equipment fault prediction [17], [18].

Although, statistical and conventional machine learning methods are comparably more transparent than deep learning

The associate editor coordinating the review of this manuscript and approving it for publication was Huiyu Zhou.

methods, in recent years, deep learning has shown breakthroughs in fault prediction [3]. Easier feature extraction, more power in pattern extraction, and multi-modal analysis make deep learning more widespread [19], [20]. Their popularity and success have led to their pervasive use in predictive preventive maintenance. [21] proposed a deep belief networks ensemble algorithm by incorporating several training techniques for predicting equipment's remaining useful life. [22] reported the use of deep fuzzy echo state networks and deep hybrid state networks for equipment fault detection. [18] also proposed a batch-normalized deep neural network to extract fault features.

Among deep learning models, Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM) [23], and Gated Recurrent Units (GRU) [24], have become popular time-series processing methods [3], [25], [26].

For vibration analysis, LSTM analyzes the inherent correlation of vibration signals in the processing of time series data, so it is increasingly adopted in data-driven fault prediction [1], [3], [12], [27], [28]. [12] proposed a Convolutional Bi-directional LSTM to predict machinery faults using sensor data. [2] used the orthogonal experimental design method to optimize the model parameters and a feature engineering to remove outliers in the fault prediction. In [3], the authors proposed a method by incorporating multiple sparse auto-encoders with LSTM for predicting mechanical faults. [1] also used vibration sensors and LSTM for fault prediction. Some research works tried to improve the performance of LSTM by adding a Convolutional Neural Network (CNN). They have shown that the combination can boost the feature selection capability of LSTM. [29] proposed a system schema that integrates programmable logic controller signals with sensor signals for real-time prediction of remaining useful life in machining tools; they added a one-dimensional CNN to LSTM. [13] introduced a hybrid model that combines a CNN with a stacked bi-directional and unidirectional LSTM network to address sequence data in the task of tool remaining useful life prediction.

Despite the success of deep learning in many applications, they are often criticized for being black-box and lacking transparency due to their non-linear multilayer structures [30]–[33]. When a model is considered a black box, its internal processes are either unknown or are known but uninterpretable by humans. In this situation, Explainable Artificial Intelligence (XAI) methods are usually adopted to explain such black-box machine learning models [34]. XAI methods aim to interpret how a deep neural network operates through techniques highlighting some model parts [32]. To interpret means providing the meaning or explaining and demonstrating a complex concept to a human [31], [35]. Interpretability has different aspects, including the necessity of interpretation, complete versus partial understanding, user's availability time to adopt, and the user's background knowledge [31]. To assess the performance of an interpreting model, several aspects should be taken into consideration, including interpretability: how much the model or its results are understandable by humans [36]; accuracy: to what extent the model precisely predicts unseen instances; fidelity: how much the model can imitate the results of a black-box baseline predictor [31].

Post hoc explanation techniques can be categorized as text explanations, visualizations, local explanations, explanations by example, explanations by simplification, and feature relevance [30]. Text explanations provide interpretability for an opaque model by learning to generate explaining texts. Visualization inclines to depict the model's behavior. Local explanations segregate the solution space and explain less complex subspaces relevant to the whole model. Explanations by example consider extracting data examples that relate to the result generated by a model. Explanations by simplification highlight those methods in which an entirely new system is recreated upon the baseline model [30]. Finally, feature relevance highlights the relation between the model output and most contributing features [30], [33], [37].

As the approach of this paper is to explain a black-box model through an instance-wise feature selection, these techniques have been described in more depth. Although finding a global subset of relevant features is a well-studied problem and has existing solutions [38], [39], they consider the same features for all samples [33]. On the other hand, instance-wise feature selection highlights essential features for each individual separately. Therefore, it is not as straight forward as the global feature selection techniques. In the literature, two streams deal with instance-wise feature selection. The first category computes the gradient of the output of the correct class concerning the input vector for the given model and uses it as a saliency map for highlighting the relevant features of the input [40]–[43]. For example, [40] proposed a method using a Parzen window approximator to compute the gradient of an opaque classifier. In [41], the authors provided ''visual explanations'' for decisions from a class of CNN-based models. Their method Gradient-weighted Class Activation Mapping (Grad-CAM) applies the gradients of any target class sent through the final convolutional layer to produce a coarse localization map denoting the critical regions in the image.

The other group approximates the model to be interpreted via a locally additive model to explain the difference between the model output and some ''reference'' output in terms of the difference between the input and some ''reference'' input [33], [37], [44]–[48]. [44] proposes Layer wise Relevance Propagation (LRP). LRP is a model-dependent technique that redistributes the predictions to each neural network layer until assigning a particular relevance score to each neuron. Although it is model dependent, [49] proposed an LRP-based model to select the most participating features in LSTM. [47] introduced an algorithm called LIME, which can explain predictions of any model by approximation with more straightforward and more interpretable linear local models around a data point. [48] presented DeepLIFT, a method designed specifically for neural networks, which decomposes the output on a specific input by backpropagating the contribution to every feature. [46] used Shapley values to measure the importance of the features of a given input and proposed a sampling-based method, ''kernel SHAP'' for estimating

TABLE 1. Comparison between instance wise feature selection methods from three different aspects. Efficiency assesses the computational times during a single explanation [37]; model free indicates whether a method is generic [37]; and number of relevant features shows whether a model requires a predefined number of features [33].

Method	Key ideas	Efficiency	Model free	Predefined feature number
Parzen [33]	Parzen window approximation of the original model.	High	Yes	Yes
Salient map [43]	The gradient of the correct class's output regarding the input.	High	No	Yes
LRP [37]	Tracking the backpropagation reversely.	High	No	Yes
Lime [40]	Locally linear approximations	Low	Yes	Yes
Kernel SHAP [39]	Shapley values to quantify the importance of features.	Low	Yes	Yes
DeepLIFT [41]	Unpacking the output of a neural network on a specific input by backpropagating the contribution to features.	High	No	Yes
L2X [30]	Mutual Information maximization with Gumbel-SoftMax trick.	High	Yes	Yes
INVASE [26]	Minimize KL using deep NN influenced by actor-critic model.	High	Ye	No
GSX (proposed)	Minimizing the error with a Gumbel-Sigmoid layer.	High	Yes	No

Shapley values to estimate the finite difference between an input vector and a reference vector.

[37] came up with a different approach and introduced learning to explain (L2X). Their method is based on learning a distinct function to extract a subset of the most informative features for each given example. This feature selector is trained to maximize the mutual information between selected features and the response variable. [33] reformulated the problem by adopting an actor-critic model to harness the problem of backpropagating through subset sampling. Their solution consists of three neural networks, a black-box baseline model, a selector to segregate a feature subset, and an evaluator which accepts a sparse input and generates the same conditional distribution as the baseline. The cost function adopted in this work is the Kullback-Leibler (KL) divergence between the full conditional distribution and the selected-features-only [33]. Table 1 compares the instance-wise explanation techniques.

In several use cases, there will undoubtedly be applications where methods such as HMM or Bayesian methods will work well or even be better than use of deep neural networks. Our focus is on those cases where deep neural networks are adopted and where greater transparency is needed. To this end the current work evaluates several recurrent deep learning models for fault prediction in disintegrated bearings using vibration sensor readings. This is a type of univariate time series analysis problem. The model aims to predict the next condition fault/normal of the bearings within the time window ahead. After building and evaluating the predictors, a novel model agnostic instance-wise interpretation method has been presented. The proposed model uses the Gumbel-Sigmoid trick [50], namely GSX, to identify the most important readings in a particular input sequence for the baseline model's prediction. In GSX, we embedded an automated instance-wise feature selection with deep learning models. Therefore, in addition, to comparing some deep learning models for fault prediction, the most important novelty of this paper is on providing further explanation for deep learning models. We compared GSX with two state-of-the-art techniques. In addition to the research dataset, a public dataset is used to validate results of the models and techniques. To the best of our knowledge, it is the first time that such a method has been adopted.

The remaining paper is organized as follows. Section II provides the research framework and presents GSX. Section III describes the models' configurations, the dataset, and the given experimental results. Section IV concludes the paper and provides new directions to continue this research.

II. FRAMEWORK

The first goal of this research is to predict a machine fault within the time window ahead. This binary classification task predicts whether a sequence of vibration readings leads to a fault or a normal condition. Therefore, the annotated dataset D is as follows:

$$D = \{(x_1, y_1), \cdots, (x_n, y_n)\}$$

In this dataset $x_i \in R^T$ is the input sequence, T is the time window size. And $y_i \in \{0, 1\}$ is the binary label in which 0 denotes the normal condition and 1 as the fault. The prediction model $f^\gamma : x \longrightarrow [0, 1]$ parametrized by γ, as the baseline, is a deep learning model which undertakes the binary classification. This model will be trained using a subset of D and tested using the rest.

The second goal is to explain the baseline model f^γ through a proposed instance-wise feature selection. Let's consider m (in this paper, we use m and T interchangeably as the number of features and the size of time window respectively) as the number of original features; the aim is to identify which features $d \subseteq m$ contribute most to the baseline output for each instance. Regarding the generic form of the model's input and output, let $x \in R^m$ be an m dimensional input and $y \in \{1, \cdots, C\}$ be the related discrete labels containing C classes. The goal is to find $S \in \{0, 1\}^m$ as the selection vector in which $s_i = 1$ indicates the ith feature has been selected, and $s_i = 0$ demonstrates it is omitted. Applying this selection vector on the original input x, leads to x^S as the suppressed feature vector:

$$x_i^S = \begin{cases} x_i & \text{if } s_i = 1 \\ * & \text{if } s_i = 0 \end{cases}$$

Asterix (*) denotes the feature is not selected. Therefore, if the output of the baseline is $\hat{y}|x$, the goal can be summarized to find the smallest selected features, S, so that:

$$\mathbb{E}(\hat{y}|x^S) \cong \mathbb{E}(\hat{y}|x) \tag{1}$$

Equation (1) denotes the conditional distribution of $\hat{y}$ given the selected features x^S be as close as possible to $\hat{y}$ having all features.

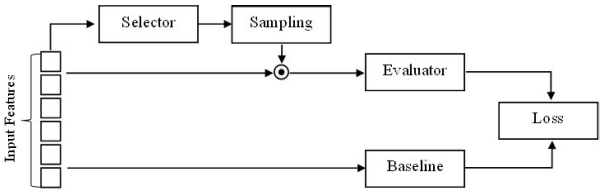

FIGURE 1. The main process of the instance-wise feature selection in which a selector learns to select the most important features for any individual input, and an evaluator assesses the performance of the new feature subset against the baseline model, which uses the all features.

One way to estimate and evaluate the selection vector S for each instance is to provide two auxiliary models [33], [37]. The selector model (2), parametrized by θ, accepts an input and generates m independent probabilities with the same size as the input features.

$$\pi_\theta : x \longrightarrow [0, 1]^m \tag{2}$$

So far, $P \in [0, 1]^m$ is the output of π_θ and shows the importance of each feature, yet a selection method is needed to make it sparse to have the selection vector S. One can simply consider these m probabilities as success rates and apply $Bernoulli: [0, 1]^m \longrightarrow \{0, 1\}^m$ to estimate S. The evaluator (3), parametrized by φ, accepts x^S (the dot product of the input and the selection vector resulted by $\pi_\theta, x \odot S$) and undertakes the same task as the baseline, aiming to evaluate the selector's performance and generating C (as many as the number of classes) probabilities.

$$f^\varphi : x^S \longrightarrow [0, 1]^C \tag{3}$$

An appropriate loss function L should be considered to measure the difference between the evaluator and the baseline outputs. A method is also needed to hold the cardinality of S as minimum as possible. Fig 1 summarizes the described process for instance-wise feature selection.

The selector, the evaluator, and the baseline can be any model such as CNN, LSTM, etc. The evaluator and the baseline models can be trained with a cross-entropy or binary cross-entropy loss function and optimization algorithms such as gradient descent. The problem is to train π_θ concerning the L (the loss function measuring the difference between f^φ and f^γ). The connection between π_θ and L has been discretized by the sampling mechanism; consequently, it is non-differentiable, and the gradient $\nabla_\theta L$ cannot be calculated and backpropagated to train π_θ.

Wrapping all the above together, to come up with an appropriate solution, one needs to find; (i) an appropriate loss function to satisfy (1), (ii) a way to train the selection model, and (iii) a technique to minimize the number of selected features.

Since the proposed solution takes advantage of L2X [37] and INVASE [33], before going ahead with the proposed method, these two state-of-the-art techniques are described in the below.

[37] used a variational lower bound of the mutual information between the output distribution of the baseline and the evaluator.

$$\max_{\theta,\varphi} \mathbb{E}_{x,y,\zeta}[\log g_\varphi(V(\theta, \zeta) \odot x, y)] \tag{4}$$

In (4) x and y are model's input and output, respectively, $g_\varphi(.)$ parametrized by φ is the proposed evaluator, which accepts the x^S containing the selected features and generating the given result in a conditional distribution. In the original works [37] the evaluator is a neural network. The cardinality of $V(\theta, \zeta)$ which is the selector part, in this work is fixed. In L2X, researchers applied the Gumbel-SoftMax trick [51] to make $V(\theta, \zeta)$ concrete as a representation for the selection vector S, yet differentiable. Empirically this trick has shown a lower variance than REINFORCE applied in works such as [33], [52]. Gumbel-SoftMax applies the concrete distribution from a categorical distribution [53] as a continuously differentiable approximation to the categorical distribution. So, it could be a candidate solution for the problem with the selector model π_θ. When one wants to approximate a categorical random variable with category probability $P = (p_1, p_2, \ldots, p_m)$, an independent random perturbation from a $Gumbel(0, 1)$ distribution can be generated as (5).

$$G_i = -\log[-\log(u_i)], \quad u_i \sim Uniform(0, 1) \tag{5}$$

Then this Gumbel noise will be added to the log probability of each category and have a temperature-dependent SoftMax over an m dimensional vector. It should be noted that $\tau \geq 1$ is the temperature and a hyperparameter to control the level of concreteness in the resulted vector.

$$K_i = \frac{e^{\frac{\log p_i + G_i}{\tau}}}{\sum_{j=1}^{m} e^{\frac{\log p_j + G_j}{\tau}}} \tag{6}$$

The resulting random vector $K = (k_1, k_2, \ldots, k_m)$ from (6) is the concrete transformation of P. Therefore, in L2X, they applied this trick to approximate the subset sampling, aimed to select d features from m ones and to calculate $S = \{s_i \in \{0, 1\}^m | \sum s_i = d\}$. Concretely $\pi_\theta : R^m \longrightarrow R^m$ has been defined to map the input with the same dimensional vector, where the ith entry of $\pi_\theta(x)$ denotes the importance score of the ith feature. They sampled a single feature out of m features independently for d times through the mentioned trick. Such a scheme samples most important d features and is easier to approximate by a continuous relaxation. Finally, a d-dimensional random vector V has been defined that is the element-wise maximum of $C^1, \ldots, C^d$:

$$K^j \sim Concrete(\pi_\theta), \quad j = 1, \ldots, d$$
$$V = (V_1, \ldots, V_m), \quad V_i = \max_j K_i^j$$

The random vector V is then used to approximate the d-hot random vector S during training. So, the authors transformed π_θ to $V(\theta, \zeta)$ as V is a function of θ and a collection of auxiliary random variables ζ sampled independently from the Gumbel distribution. Then they applied the elementwise product $V(\theta, \zeta) \odot x$ as an approximation of x^S [37]. The performance of L2X is good both during the training and the explanation [37]. However, due to the Gumbel-SoftMax,

the number of selected features must be fixed in advance as an extra hyperparameter and is necessarily fixed for all instances, whereas in reality, every example has a different number of essential features [33].

Authors of [33] continued the work of L2X and tackled its mentioned problem by proposing INVASE. They used *KL* divergence between the conditional distributions $\hat{y}|x$ and $\hat{y}|x^S$ to satisfy equation (1) instead of the mutual information used in [37]. They proposed an actor-critic model [54] to bypass backpropagation through the sampling and instead used the predictor network to provide a reward to the selector network. Instead of a fixed number of features for all instances, the authors used an l_0 regularization term for the selector's outputs controlled by a hyperparameter ($\lambda \geq 0$) [33]. The bigger (λ results to fewer selected features and vice versa. Finally, they train three distinct models as follows.

Such as (2), the selector is $\pi_\theta : R_m \longrightarrow R^m$ followed by a *Bernoulli* sampler to estimate S or the selection vector. x^S ($S \odot x$) is the input of the evaluator model $g_\varphi(x^S, y)$. As mentioned, they train all three models (selector, evaluator, and baseline) together.

$$l(\varphi) = -\mathbb{E}_{(x,y)\sim p, S\sim\pi_\theta(x,.)}[\sum y_i \ln g_i^\varphi(x^s)] \tag{7}$$

$$l(\gamma) = -\mathbb{E}_{(x,y)\sim p}[\sum y_i \ln f_i^\gamma(x)] \tag{8}$$

Equations (7) and (8) are cross-entropy loss functions to train the evaluator and the baseline, respectively. To prepare the selector model, they considered a reward, which is the *KL* difference between the evaluator and the baseline (9).

$$\hat{l}(x, s) = -[l(\varphi) - l(\gamma)] \tag{9}$$

The authors combined a binary cross-entropy function between the output of π_θ and the selection vector S, multiplied by the reward $\hat{l}(x, s)$), along with the l_0 norm of the selector output came up with (10).

$$l(\theta) = \mathbb{E}[(\sum x_i^S \ln \hat{S}_i^\theta + x_i^S \ln(1 - \hat{S}_i^\theta))(\hat{l}(x, s)) + \lambda \parallel S \parallel)] \tag{10}$$

This technique is still model agnostic and could resolve the fixed number of selected features problem with L2X. Its performance during the explanation is good; however, since it is a Reinforcement learning model, its convergence time during the training is high, and it is sensitive to hyper-parameters such as learning rates [37].

In the proposed method, we have considered the problems with L2X (fixed selected features number) and INVASE (performance) and developed solutions to both. In order to articulate the selector and evaluator with a continuous but still concrete connection, instead of using Gumbel-SoftMax, we used the Gumbel-Sigmoid proposed in [50]. The Gumbel-Sigmoid implementation is like Gumbel-SoftMax [51] to select a subset of features from all features contributing more to the final prediction. However, there are two differences between these two methods; in Gumbel-Sigmoid (i) there are two uniform noises (11), and (ii) these noises are added to the sigmoid function (12) instead of a SoftMax function.

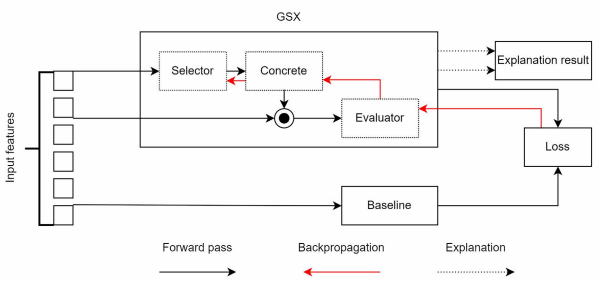

FIGURE 2. The GSX workflow. During the training, data passes through GSX and the baseline model simultaneously to calculate the error. Then, the error and a regularization penalty backpropagate through GSX. In this model, the selector, followed by a Gumbel-Sigmoid layer (concrete layer), and the evaluator, forms an integrated deep learning model. During the explanation, GSX provides two outputs, the prediction result and a sparse vector denoting the essential features of the given instance.

$$G_i = -\log(\frac{\log u_i^1}{\log u_i^2}), \quad u_i^1, u_i^2 \sim Uniform(0, 1) \tag{11}$$

$$K_i = \frac{1}{1 + e^{-\frac{(p_i + G_i)}{\tau}}} \tag{12}$$

In equation (12), the level of concreteness is controlled by the temperature hyperparameter ($\tau \geq 1$) in which $\tau = 1$ makes it to a simple Sigmoid, whereas a higher value of τ, leads to a more concrete result. Therefore, the output of $\pi_\theta(X)$ passes through a layer of Gumbel-Sigmoid units of the same size.

$$V(\theta, \zeta_1, \zeta_2) : x \longrightarrow Concrete^m \tag{13}$$

Equation (13) denotes the new selector parametrized by θ and a collection of two set auxiliary random variables ζ_1 and ζ_2 sampled independently from the Gumbel distribution. The dot product of the selector's output and the input features passes through the next part as the evaluator (14).

$$g_\varphi(V(\theta, \zeta_1, \zeta_2) \odot X, Y) : x^S \longrightarrow [0, 1]^c \tag{14}$$

Finally, such as INVASE, a l_0 regularization term of the output of (13) is added; therefore, the loss function becomes (15).

$$\max_{\theta,\varphi} \mathbb{E}_{x,y,\zeta_1,\zeta_2}[\log g_\varphi(V(\theta, \zeta_1, \zeta_2) \odot x, y) + \lambda \parallel V(\theta, \zeta_1, \zeta_2) \parallel] \tag{15}$$

In this loss function, $\lambda \geq 0$ controls the number of selected features (such as INVASE, bigger λ means fewer selected features and vice versa). It should be noted that, during the training, data pass through the model, and the error is back-propagated using the concrete transformation. But the selector (13) output will be rounded to provide the given selection vector during the explanation. Fig 2 summarizes the GSX training and explanation process.

III. EXPERIMENTS

We conducted experiments on two datasets. The first dataset is provided through the project of this research. And the

second one is a public dataset for electrical fault detection and classification [55], [56] through which we validated the models of the research. We carried out empirical experiments to examine RNN, LSTM, GRU, CNN, LSTM-GMP, CNN-LSTM, and CNN-GRU on both datasets. Then by selecting one of the trained models as the baseline, the performance of GSX is compared with L2X and INVASE. We use ADAM [57] for all experiments with the default hyper-parameters. The temperature for Gumbel-SoftMax and the Gumbel-Sigmoid approximation are fixed to be 0.1 [37]. Codes and the dataset are available in the following:

- GSX: https://github.com/tahamsi/gsx.
- INVASE: https://github.com/jsyoon0823/ INVASE.
- L2X: https://github.com/Jianbo-Lab/L2X.

All codes have been run on a machine with the following specifications:

- OS: Microsoft Windows 10 Pro X64.
- Processor: Intel® Core I7, CPU 1.8 GHz.
- Memory: 16 GB.

A. INVISIBLE SYSTEMS LTD DATASET

We received four months of vibration readings from four industrial bearings extracted from Invisible Systems Ltd's web platform to prepare the dataset. All bearings have the same size, class, and category (Size: 0.55 KW, Class: I, and Category B2). Based on ISO 10816, the vibration threshold for this type is 0.5. One of these bearings was disintegrated and has been faulty many times, and the others were in respectively better condition having either no or a few faults. Since the problem is a univariate classification task, the dataset is annotated for different window sizes and available on https://github.com/tahamsi/gsx/dataset.

1) PREDICTION

The data is split into a train set and a test set to predict a fault happening within the next time window. Several deep learning models were developed, including RNN, LSTM, GRU, LSTM, LSTM-GMP, CNN, CNN-LSTM, and CNN-GRU. To carry out the experiments, many topologies consisting of the number of neurons and the number of dense layers is examined for each, and the best configuration based on these experiments is selected. Fig 3 displays LSTM's training loss as an example Adam's performance with the given learning rate.

Six scenarios have been designed based on different time window sizes to investigate the performance of the models.

Fig 4 implies that almost all predictors could predict the fault. In scenarios with the smaller time window size, the performances of the predictors are very close, whereas, in the bigger window size, RNN shows inferior performance due to the vanishing gradient [58]. Finally, combined methods such as CNN-GRU outperform those experiments in which the window size is considerably large.

2) EXPLANATION

In the explanation part (instance-wise feature selection), because of simplicity to show the result, the window size

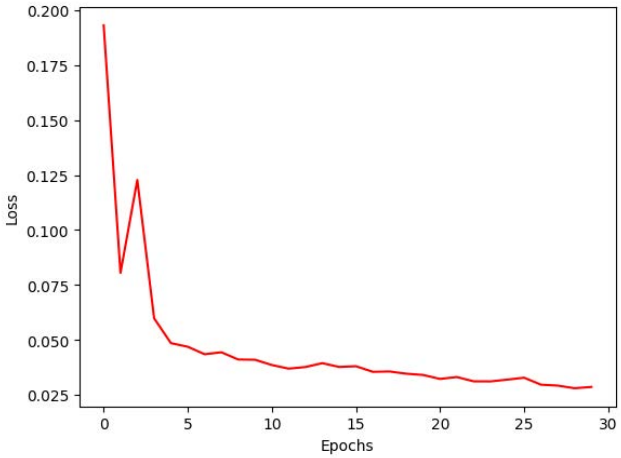

FIGURE 3. LSTM training error for window size 20. The training has been conducted with Adam optimizer, and the learning rate was 0.1.

is fixed at 20. The models' structures and hyper-parameters have been identified through a grid search.

Since the LSTM showed acceptable performance during the prediction, and it is still a black-box model, LSTM is selected as the baseline. Therefore, the structure of the baseline is (fc stands for fully connected):

$$lstm(60) \longrightarrow fc(32, relu) \longrightarrow fc(1, sigmoid).$$

In INVASE, the evaluator and baseline are the same, whilst the selector is a feedforward neural network (T is the window size)

$$(fc(T, relu) \longrightarrow fc(20, relu) \longrightarrow fc(30, relu) \longrightarrow fc(T, sigmoid)).$$

Likewise, L2X used the same evaluator and baseline, the selector is:

$$fc(T, relu) \longrightarrow fc(20, relu) \longrightarrow fc(30, relu) \longrightarrow Gumbel - SoftMax(T).$$

The temperature has been considered 0.1, the same as [37]. In GSX, the evaluator part and the baseline are the same as the other methods. The temperatures is set 0.1. The selector is:

$$fc(T, relu) \longrightarrow fc(20, relu) \longrightarrow fc(30, relu) \longrightarrow Gumbel - Sigmoid(T).$$

For L2X, the number of selected features is fixed 10 out of 20. To make the comparisons fair, different values of the regularization parameter λ for INVASE and GSX have been tested to reach the same number of selected features in average as L2X. Table 2 summarizes the average of 30 runs for each model.

As evident in table 2, the performance measures including accuracy and f-measure of all models are close to the baseline; therefore, it can be concluded that they were loyal to the baseline [31]. So, all models could explain the baseline to

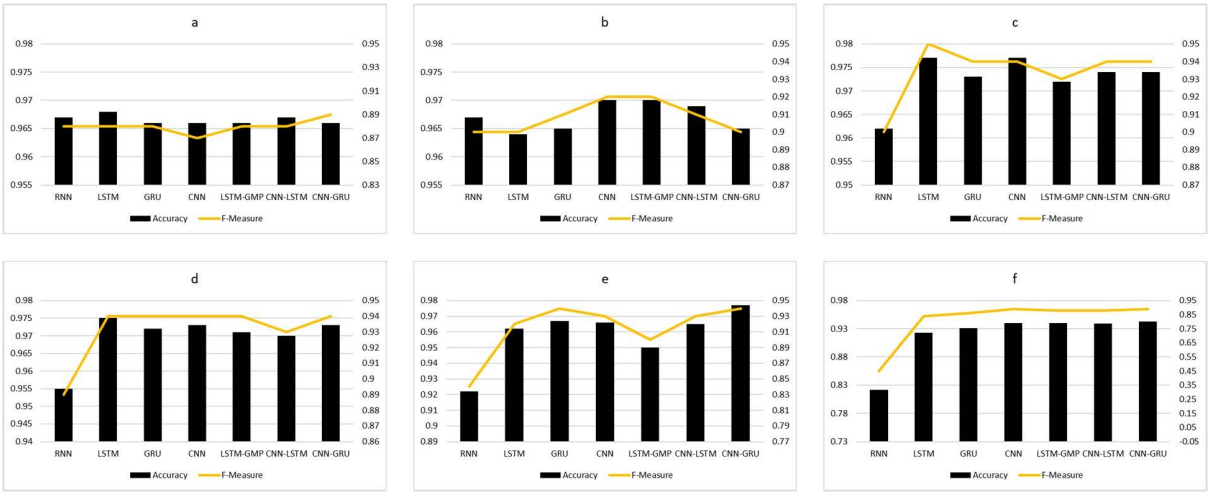

FIGURE 4. Comparing deep learning models regarding different window size. a) window size is 5, b) window size is 10, c) window size is 20, d) window size is 30, e) window size is 60, and f) window size is 120.

TABLE 2. Comparing GSX with other instance-wise feature selection techniques (L2X and INVASE) in terms of performance and run-time. The number of selected features are fixed for L2X, and controlled (by the related penalty term) in GSX and INVASE.

Model	Accuracy	F-Measure	Average # of features	Training time (s)	Explanation time (s)
Baseline	0.968	0.9	All		
L2X	0.964	0.92	10 fixed	35.6	1.1
INVASE	0.916	0.71	10	575	2.1
GSX	0.963	0.92	10	29.7	0.8

some extent. However, L2X and GSX performed better than INVASE.

On the one hand, although INVASE is a flexible method that provides variable selected features, its performance has been relatively poor compared to the others. The convergence time for INVASE was also considerably high, which is typical in reinforcement learning methods. On the other hand, L2X shows good accuracy and run times performance. Since this method considers the selector and the evaluator as a single model, its convergence and explanation times are low. Nevertheless, as mentioned earlier, in L2X, the number of selected features should be considered in advance and fixed for all instances; since this value varies in a different situations, it is a drawback. In terms of accuracy, GSX performs as well as L2X, but its training time and explanation time are slightly better than L2X. Moreover, it does not need to have a fixed number of features in advance and is as flexible as INVASE. Therefore, regarding the experiments, GSX could take advantage of both previous methods and overcome their deficiencies through a novel solution based on Gumbel with a sigmoidal combination.

Fig 5 shows an example of GSX results on two sequences containing 20 readings. As highlighted, for the normal sample members indexed 2, 4, 6, 8, 9, 10, 11, 13, 15, and 18 participated in this decision, and for the faulty sample members indexed 1, 3, 4, 6, 9, 12, 13, 18, 19, and 20 contributed in the decision.

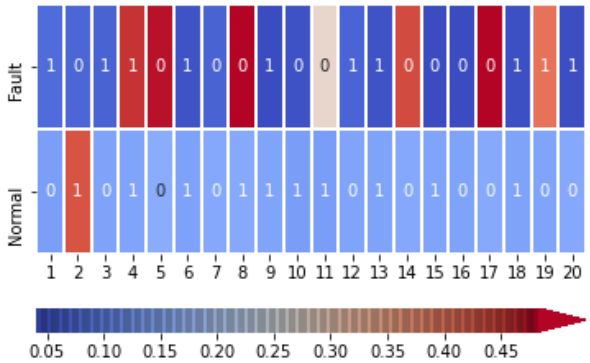

FIGURE 5. An example of GSX output. This heatmap compares two sample and their selected features. The colours depict the amount of vibration, 0 shows that the related reading is omitted and 1 shows that is selected.

B. ELECTRICAL FAULT DETECTION DATASET

The next dataset is a public dataset for the electrical fault detection and classification [55], [56] accessible via https:// www.kaggle.com /datasets/esathyaprakash/electrical-fault-detection-and-classification/detectdataset.csv. This data set is related to a power system consisting of four generators of 11×10^3 Volt, each pair located at each end of the transmission line. Transformers are present in between to simulate and study the various faults at the midpoint of the transmission line. The line voltage and line current parameters are compared with the preloaded datasets, and hence, the faults are classified. It contains six features and one class variable denoting normal or fault condition. This dataset is used to validate the models of this research.

1) PREDICTION

Like the previous dataset, the data is split into two sets to train and test the predictors. The deep learning models, including RNN, LSTM, GRU, LSTM, LSTM-GMP, CNN, CNN-LSTM, and CNN-GRU are trained on the training set.

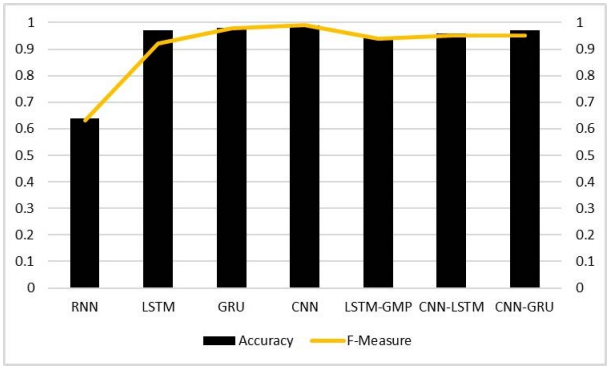

FIGURE 6. Comparing deep learning models on the electrical fault detection dataset, based on their accuracy and f-measure.

All topological parameters, such as the number of neurons and the number of dense layers, are examined, and the best configuration based on these experiments is selected. On this dataset the models learn to classify a reading into fault or normal condition, based on 6 features.

Fig 6 displays that, regarding the performance measures, all model could conduct the classification task well. However, RNN has been overfitted and showed a poor performance in this dataset.

2) EXPLANATION

For the explanation part (instance-wise feature selection), the goal is to highlight which features among the six predictors contribute most in the final classification of each instance. LSTM has been chosen as the baseline model. Through the grid search the structures of all models are identified. The structure of the baseline is as follow:

$$lstm(30) \longrightarrow fc(28, relu) \longrightarrow fc(1, sigmoid).$$

In INVASE, the selector's structure is (m is the number of features)

$$(fc(m, relu) \longrightarrow fc(5, relu) \longrightarrow fc(10, relu) \longrightarrow fc(m, sigmoid)).$$

In L2X the selector is:

$$fc(m, relu) \longrightarrow fc(8, relu) \longrightarrow fc(16, relu) \longrightarrow Gumbel - SoftMax(m).$$

The temperature has been considered 0.1, the same as [37]. In GSX, the temperatures is also 0.1. The selector is:

$$fc(m, relu) \longrightarrow fc(5, relu) \longrightarrow fc(10, relu) \longrightarrow Gumbel - Sigmoid(m).$$

The number of selected features for L2X is 4 out of 6. And such as the previous dataset to make the comparisons fair, different values of the regularization parameter λ for INVASE and GSX have been examined to identify the same number of selected features as L2X. Table 3 summarizes the average results of 30 runs for each model.

As evident in table 3, in this dataset the performance of L2X and GSX are close to the baseline; therefore, they can conduct the classification the same as the baseline. However, INVASE's performance is quit far from the baseline. Moreover, GSX performed better than L2X with better run-time.

TABLE 3. Comparing GSX with other instance-wise feature selection techniques (L2X and INVASE) in terms of performance and run-time on the electrical fault detection dataset. The number of selected features are fixed 4 out of 6 for L2X, and controlled (by the related penalty term) in GSX and INVASE.

Model	Accuracy	F-Measure	Average # of features	Training time (s)	Explanation time (s)
Baseline	0.99	0.99	All		
L2X	0.97	0.98	4 fixed	14	0.5
INVASE	0.89	0.89	4	279	0.8
GSX	0.99	0.99	4	13.6	0.4

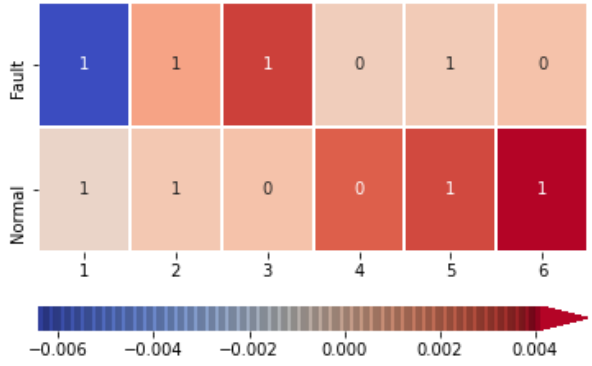

FIGURE 7. An example of GSX output on the electrical fault detection dataset. This heatmap compares two sample and their selected features. The colours depict the amount of the features, 0 shows that the related reading is omitted and 1 shows that is selected.

Fig 7 shows an example of GSX results on two readings. For the normal sample members indexed 1, 2, 5, and 6 participated in the classification result; for the faulty sample members indexed 1, 2, 3, 5 contributed in the decision.

IV. CONCLUSION

This paper develops an explainable deep learning model for the predictive preventive maintenance task, aiming to predict a fault happening in the near future by processing the preceding vibration signals. The primary dataset consists of chronological sequences of vibration readings and their related labels denoting whether they were faulty or not. We implemented and tested all models of this research upon this dataset. Furthermore, we used a public dataset for electrical fault detection to validate the findings.

Different deep learning models have been examined, including RNN, LSTM, GRU, CNN, LSTM-GMP, CNN-LSTM, and CNN-GRU. The experiments show that all considered prediction models perform well; however, the RNN's performance deteriorates with a more significant time window due to the vanishing gradient problem. The hybrid models with larger window sizes (combination of CNN with LSTM or GRU) outperformed the other models.

In order to explain the resulted prediction models, among the different post-hoc explanation methods, the paper considers the use of ''instance-wise feature selection'' methods to highlight the most contributing features for each individual

separately. Whilst deep learning models are black-box, this approach can highlight based on which features such model makes a decision for each input. To this end, different methods have been reviewed, and among them, two state-of-the-art methods, L2X [37] and INVASE [33] are investigated deeply. Both methods aim to make the selection process trainable, minimize the number of selected features, and keep the auxiliary feature selection model loyal to the baseline. A detailed review of these methods revealed that: (i) in L2X, the number of selected features is fixed for all instances, limiting its applicability in real applications, (ii) the convergence time in INVASE is high because it trains all three parts including selector, evaluator and baseline together, and as a reinforcement learning technique, its output is sensitive to the value of its hyper-parameters.

To cope with the existing problems with L2X and INVASE, this paper has developed a novel model-agnostic explainable method called Gumbel-Sigmoid eXplanator (GSX). Instead of Gumbel-SoftMax reparameterization trick applied in L2X, this method uses Gumbel-Sigmoid to make the selection process differentiable. Unlike the Gumbel-SoftMax in which the d-hot output results from the SoftMax function, and one needs to take the top d members, Gumbel-Sigmoid generates independent concrete results for each unit. To sum up, in this method (i) we have developed a differentiable and trainable connection between the selector and the evaluator, and (ii) we can flexibly control the number of selected features without relying on a predefined number of features by using the l_0 penalty term. The experimental results on both datasets indicated the outperformance of GSX against L2X and INVASE.

For future works, GSX can be extended to high-dimensional and multi-modal data such as multi-variate time series, images, text, videos, and so on. To this end, it is worth investigating the use of GSX for different deep learning models for various applications such as Computer Vision, NLP, and Signal Processing.

ACKNOWLEDGMENT

The authors would like to thank ISL for providing the dataset.

REFERENCES

[1] C. I. Nwakanma, F. B. Islam, M. P. Maharani, D.-S. Kim, and J.-M. Lee, "IoT-based vibration sensor data collection and emergency detection classification using long short term memory (LSTM)," in *Proc. Int. Conf. Artif. Intell. Inf. Commun. (ICAIIC)*, Jeju Island, South Korea, Apr. 2021, pp. 273–278, doi: 10.1109/ICAIIC51459.2021.9415228.

[2] W. Zhang, W. Guo, X. Liu, Y. Liu, J. Zhou, B. Li, Q. Lu, and S. Yang, "LSTM-based analysis of industrial IoT equipment," *IEEE Access*, vol. 6, pp. 23551–23560, 2018, doi: 10.1109/ACCESS.2018.2825538.

[3] J. Guo, Z. Lao, M. Hou, C. Li, and S. Zhang, "Mechanical fault time series prediction by using EFMSAE-LSTM neural network," *Measurement*, vol. 173, Mar. 2021, Art. no. 108566, doi: 10.1016/j.measurement.2020.108566.

[4] T. Plante, A. Nejadpak, and C. X. Yang, "Vibration analysis: Fault detection and failure prediction," in *Proc. IEEE AUTOTESTCON*, Dec. 2015, p. 5.

[5] M. Rahnama, A. Vahedi, A. M. Alikhani, and A. Montazeri, "Machine-learning approach for fault detection in brushless synchronous generator using vibration signals," *IET Sci., Meas. Technol.*, vol. 13, no. 6, pp. 852–861, Aug. 2019, doi: 10.1049/iet-smt.2018.5523.

[6] R. Li, C. Li, X. Peng, and W. Wei, "Electromagnetic vibration simulation of a 250-MW large hydropower generator with rotor eccentricity and rotor deformation," *Energies*, vol. 10, no. 12, p. 2155, Dec. 2017, doi: 10.3390/en10122155.

[7] J. B. Ali, N. Fnaiech, L. Saidi, B. Chebel-Morello, and F. Fnaiech, "Application of empirical mode decomposition and artificial neural network for automatic bearing fault diagnosis based on vibration signals," *Appl. Acoust.*, vol. 89, pp. 16–27, Mar. 2015, doi: 10.1016/j.apacoust.2014.08.016.

[8] M. T. Pham, J.-M. Kim, and C. H. Kim, "Accurate bearing fault diagnosis under variable shaft speed using convolutional neural networks and vibration spectrogram," *Appl. Sci.*, vol. 10, no. 18, p. 6385, Sep. 2020, doi: 10.3390/app10186385.

[9] T. Chen, Z. Wang, X. Yang, and K. Jiang, "A deep capsule neural network with stochastic delta rule for bearing fault diagnosis on raw vibration signals," *Measurement*, vol. 148, Dec. 2019, Art. no. 106857, doi: 10.1016/j.measurement.2019.106857.

[10] M. Altaf, T. Akram, M. A. Khan, M. Iqbal, M. M. I. Ch, and C.-H. Hsu, "A new statistical features based approach for bearing fault diagnosis using vibration signals," *Sensors*, vol. 22, no. 5, p. 2012, Mar. 2022, doi: 10.3390/s22052012.

[11] S. R. Saufi, Z. A. B. Ahmad, M. S. Leong, and M. H. Lim, "Low-speed bearing fault diagnosis based on ArSSAE model using acoustic emission and vibration signals," *IEEE Access*, vol. 7, pp. 46885–46897, 2019, doi: 10.1109/ACCESS.2019.2909756.

[12] R. Zhao, R. Yan, J. Wang, and K. Mao, "Learning to monitor machine health with convolutional Bi-directional LSTM networks," *Sensors*, vol. 17, no. 2, p. 273, Jan. 2017, doi: 10.3390/s17020273.

[13] Q. An, Z. Tao, X. Xu, M. El Mansori, and M. Chen, "A data-driven model for milling tool remaining useful life prediction with convolutional and stacked LSTM network," *Measurement*, vol. 154, Mar. 2020, Art. no. 107461, doi: 10.1016/j.measurement.2019.107461.

[14] S. Hong, Z. Zhou, E. Zio, and K. Hong, "Condition assessment for the performance degradation of bearing based on a combinatorial feature extraction method," *Digit. Signal Process.*, vol. 27, pp. 159–166, Apr. 2014, doi: 10.1016/j.dsp.2013.12.010.

[15] B. T. Narendiranath, H. S. Himamshu, K. N. Prabin, P. D. Rama, and C. Nishant, "Journal bearing fault detection based on Daubechies wavelet," *Arch. Acoust.*, vol. 42, no. 3, pp. 401–414, Sep. 2017, doi: 10.1515/aoa-2017-0042.

[16] L. Zhou, F. Duan, M. Corsar, F. Elasha, and D. Mba, "A study on helicopter main gearbox planetary bearing fault diagnosis," *Appl. Acoust.*, vol. 147, pp. 4–14, Apr. 2019, doi: 10.1016/j.apacoust.2017.12.004.

[17] L. Jianyu, S. Zhenzhong, P. M. Pardalos, H. Ying, Z. Shaohui, and L. Chuan, "A hybrid multi-objective genetic local search algorithm for the prize-collecting vehicle routing problem," *Inf. Sci.*, vol. 478, pp. 40–61, Apr. 2019, doi: 10.1016/j.ins.2018.11.006.

[18] J. Wang, J. Kang, and G. Hou, "Real-time fault repair scheme based on improved genetic algorithm," *IEEE Access*, vol. 7, pp. 35805–35815, 2019, doi: 10.1109/ACCESS.2019.2905042.

[19] B. Lim and S. Zohren, "Time-series forecasting with deep learning: A survey," *Phil. Trans. Roy. Soc. A, Math., Phys. Eng. Sci.*, vol. 379, no. 2194, Apr. 2021, Art. no. 20200209, doi: 10.1098/rsta.2020.0209.

[20] S. Razavi, "Deep learning, explained: Fundamentals, explainability, and bridgeability to process-based modelling," *Environ. Model. Softw.*, vol. 144, Oct. 2021, Art. no. 105159, doi: 10.1016/j.envsoft.2021.105159.

[21] C. Zhang, P. Lim, A. K. Qin, and K. C. Tan, "Multiobjective deep belief networks ensemble for remaining useful life estimation in prognostics," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp. 2306–2318, Oct. 2017, doi: 10.1109/TNNLS.2016.2582798.

[22] S. Zhang, Z. Sun, M. Wang, J. Long, Y. Bai, and C. Li, "Deep fuzzy echo state networks for machinery fault diagnosis," *IEEE Trans. Fuzzy Syst.*, vol. 28, no. 7, pp. 1205–1218, Jul. 2019, doi: 10.1109/TFUZZ.2019.2914617.

[23] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[24] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder–decoder for statistical machine translation," 2014, *arXiv:1406.1078*.

[25] F. Okubo, T. Yamashita, A. Shimada, and H. Ogata, "A neural network approach for students' performance prediction," in *Proc. 7th Int. Learn. Analytics Knowl. Conf.*, Vancouver, BC, Canada, Mar. 2017, pp. 598–599, doi: 10.1145/3027385.3029479.

[26] J. Wang, J. Yan, C. Li, R. X. Gao, and R. Zhao, "Deep heterogeneous GRU model for predictive analytics in smart manufacturing: Application to tool wear prediction," *Comput. Ind.*, vol. 111, pp. 1–14, Oct. 2019, doi: 10.1016/j.compind.2019.06.001.
[27] D. Huang, Y. Fu, N. Qin, and S. Gao, "Fault diagnosis of high-speed train bogie based on LSTM neural network," *Sci. China Inf. Sci.*, vol. 64, no. 1, Jan. 2021, Art. no. 119203, doi: 10.1007/s11432-018-9543-8.
[28] S. Zheng, K. Ristovski, A. Farahat, and C. Gupta, "Long short-term memory network for remaining useful life estimation," in *Proc. IEEE Int. Conf. Prognostics Health Manage. (ICPHM)*, Dallas, TX, USA, Jun. 2017, pp. 88–95, doi: 10.1109/ICPHM.2017.7998311.
[29] J. Niu, C. Liu, L. Zhang, and Y. Liao, "Remaining useful life prediction of machining tools by 1D-CNN LSTM network," in *Proc. IEEE Symp. Ser. Comput. Intell. (SSCI)*, Xiamen, China, Dec. 2019, pp. 1056–1063, doi: 10.1109/SSCI44817.2019.9002993.
[30] A. B. Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-López, D. Molina, R. Benjamins, R. Chatila, and F. Herrera, "Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI," 2019, *arXiv:1910.10045*.
[31] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi, "A survey of methods for explaining black box models," *ACM Comput. Surv.*, vol. 51, no. 5, pp. 1–42, Sep. 2019, doi: 10.1145/3236009.
[32] Y.-S. Lin, W.-C. Lee, and Z. B. Celik, "What do you see? Evaluation of explainable artificial intelligence (XAI) interpretability through neural backdoors," 2020, *arXiv:2009.10639*.
[33] J. Yoon, J. Jordon, and M. van der Schaar, "INVASE: Instance-wise variable selection using neural networks," in *Proc. ICLR*, 2019, p. 24.
[34] U. Schlegel, H. Arnout, M. El-Assady, D. Oelke, and D. A. Keim, "Towards a rigorous evaluation of XAI methods on time series," 2019, *arXiv:1909.07082*.
[35] F. Doshi-Velez and B. Kim, "Towards a rigorous science of interpretable machine learning," 2017, *arXiv:1702.08608*.
[36] A. A. Freitas, "Comprehensible classification models—A position paper," *ACM SIGKDD Explor. Newslett.*, vol. 15, no. 1, p. 10, 2013.
[37] J. Chen, L. Song, M. J. Wainwright, and M. I. Jordan, "Learning to explain: An information-theoretic perspective on model interpretation," 2018, *arXiv:1802.07814*.
[38] E. Candes, Y. Fan, L. Janson, and J. Lv, "Panning for gold: Model-X knockoffs for high-dimensional controlled variable selection," 2016, *arXiv:1610.02351*.
[39] H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 8, pp. 1226–1238, Aug. 2005, doi: 10.1109/TPAMI.2005.159.
[40] D. Baehrens, T. Schroeter, S. Harmeling, M. Kawanabe, K. Hansen, and K. R. Müller, "How to explain individual classification decisions," *J. Mach. Learn. Res.*, vol. 11, pp. 1803–1831, 2010.
[41] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual explanations from deep networks via gradient-based localization," *Int. J. Comput. Vis.*, vol. 128, no. 2, pp. 336–359, Feb. 2020, doi: 10.1007/s11263-019-01228-7.
[42] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," 2013, *arXiv:1312.6034*.
[43] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for simplicity: The all convolutional net," 2014, *arXiv:1412.6806*.
[44] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek, "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation," *PLoS ONE*, vol. 10, Jul. 2015, Art. no. e0130140.
[45] P.-J. Kindermans, K. Schütt, K.-R. Müller, and S. Dähne, "Investigating the influence of noise and distractors on the interpretation of neural networks," 2016, *arXiv:1611.07270*.
[46] S. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," 2017, *arXiv:1705.07874*.
[47] M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why should I trust you?' explaining the predictions of any classifier," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, San Francisco, CA, USA, Aug. 2016, pp. 1135–1144, doi: 10.1145/2939672.2939778.
[48] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," 2017, *arXiv:1704.02685*.
[49] A. Patil, A. Wadekar, T. Gupta, R. Vijan, and F. Kazi, "Explainable LSTM model for anomaly detection in HDFS log file using layerwise relevance propagation," in *Proc. IEEE Bombay Sect. Signature Conf. (IBSSC)*, Mumbai, India, Jul. 2019, pp. 1–6, doi: 10.1109/IBSSC47189.2019.8973044.
[50] X. Geng, L. Wang, X. Wang, B. Qin, T. Liu, and Z. Tu, "How does selective mechanism improve self-attention networks?" 2020, *arXiv:2005.00979*.
[51] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," 2016, *arXiv:1611.01144*.
[52] C. Raffel, M.-T. Luong, P. J. Liu, R. J. Weiss, and D. Eck, "Online and linear-time attention by enforcing monotonic alignments," 2017, *arXiv:1704.00784*.
[53] E. J. Gumbel, "The maxima of the mean largest value and of the range," *Ann. Math. Statist.*, vol. 25, no. 1, pp. 76–84, Mar. 1954.
[54] J. Peters and S. Schaal, "Natural actor-critic," *Neurocomputing*, vol. 71, nos. 7–9, pp. 1180–1190, Mar. 2008, doi: 10.1016/j.neucom.2007.11.026.
[55] M. Jamil, S. K. Sharma, and R. Singh, "Fault detection and classification in electrical power transmission system using artificial neural network," *SpringerPlus*, vol. 4, no. 1, p. 334, Dec. 2015, doi: 10.1186/s40064-015-1080-x.
[56] K. Janarthanam, P. Kamalesh, and T. V. Basil, "Electrical faults-detection and classification using machine learning," in *Proc. Int. Conf. Electron. Renew. Syst. (ICEARS)*, Tuticorin, India, Mar. 2022, pp. 1289–1295, doi: 10.1109/ICEARS53579.2022.9751897.
[57] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.
[58] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Trans. Neural Netw.*, vol. 5, no. 2, pp. 157–166, Mar. 1994, doi: 10.1109/72.279181.

TAHA MANSOURI (Member, IEEE) received the Ph.D. degree in information technology from Allameh Tabataba'i University, Iran, in 2016. He is currently pursuing the Ph.D. degree in artificial intelligence with the University of Salford, U.K. He is also a Lecturer in computer science (AI) with the University of Salford, where he used to be a Research Associate in the IoT. He is a Professional Member of the British Computer Society. He has published in acclaimed journals, such as the *Knowledge Based Systems*, *Expert Systems with Applications*, *Applied Soft Computing*, *Optic Letters*, and *Telecommunication Systems*. His research interests include machine learning, deep learning, reinforcement learning, and computer vision.

SUNIL VADERA received the Ph.D. degree in computer science from The University of Manchester, in 1992. He is currently a Professor in computer science with the University of Salford, U.K., where he has served in many leadership roles, including as the Dean and the Head of the School of Computing, Science and Engineering, from 2011 to 2019. He is a fellow of the British Computer Society, a Chartered Engineer (C.Eng.), and a Chartered IT Professional (CITP). He was awarded the U.K. BDO Best Indian Scientist and Engineer in recognition of his contributions to computing, science, and engineering in U.K., in 2014. He was the Chair of the U.K. BCS Knowledge Discovery and Data Mining Symposium, Salford, in 2009. He has been the General Chair of numerous conferences, including the IFIP Conference on Intelligent Information Processing, in 2010, 2012, 2014, and 2016. He was the Organizing Chair of a workshop on Cost sensitive learning at the IEEE International Conference on Data Mining, in 2012. He is passionate about closing the gap between theory and application of AI and leads the University of Salford's contribution to the Greater Manchester AI Foundry Project that supports the development of innovative AI applications by SMEs.