Layer-Wise Relevance Propagation for Interpreting LSTM-RNN Decisions in Predictive Maintenance

Haiyue Wu (wu1254@purdue.edu)
Purdue University College of Engineering
Aihua Huang
Purdue University College of Engineering
John W. Sutherland
Purdue University College of Engineering

Research Article

Keywords: Predictive maintenance, long short-term memory, Recurrent neural network, layer-wise relevance propagation, model interpretation

Posted Date: May 24th, 2021

DOI: https://doi.org/10.21203/rs.3.rs-522677/v1

Version of Record: A version of this preprint was published at The International Journal of Advanced Manufacturing Technology on September 11th, 2021. See the published version at https://doi.org/10.1007/s00170-021-07911-9.

Layer-Wise Relevance Propagation for Interpreting LSTM-RNN Decisions in Predictive Maintenance

Haiyue Wu[a,*], Aihua Huang[a], John W. Sutherland[a]

[a]Environmental and Ecological Engineering
Purdue University

500 Central Drive, West Lafayette, IN 47907, USA

wu1254@purdue.edu

Abstract

Predictive maintenance (PdM) is an advanced technique to predict the time to failure (TTF) of a system. PdM collects sensor data on the health of a system, processes the information using data analytics, and then establishes data-driven models that can forecast system failure. Deep neural networks are increasingly being used as these data-driven models owing to their high predictive accuracy and efficiency. However, deep neural networks are often criticized as being "black boxes," which owing to their multi-layered and non-linear structure provide little insight into the underlying physics of the system being monitored, and that are nontransparent and untraceable in their predictions. In order to address this issue, the layer-wise relevance propagation (LRP) technique is applied to analyze a long short-term memory (LSTM) recurrent neural network (RNN) model. The proposed method is demonstrated and validated for a bearing health monitoring study based on vibration data. The obtained LRP results provide insights into how the model "learns" from the input data and demonstrate the distribution of contribution/relevance to the neural network classification in the input space. In addition, comparisons are made with gradient-based sensitivity analysis to show the power of LRP in interpreting RNN models. The LRP is proved to have promising potential in interpreting deep neural network models and improving model accuracy and efficiency for PdM.

Keywords

Predictive maintenance; long short-term memory; Recurrent neural network; layer-wise relevance propagation; model interpretation

1 Introduction

In the era of Industry 4.0, the Industrial Internet of Things (IIoTs), smart manufacturing, and cyber physical systems [1] are providing stimulus to manufacturing sectors where smart sensors, artificial intelligence, etc. are employed to improve manufacturing performance. As an advanced maintenance technique, predictive maintenance (PdM), also known as prognostics and health management (PHM), is broadly applied to monitor machine health to improve productivity, safety and reliability, and reduce costs and waste. PdM not only provides diagnostics information on what is wrong and where the problem is, but also gives prognostics information on when and what failure is going to happen. An effective PdM system can detect incipient failure or fault and then schedule maintenance procedures at an early stage.

The concept of PdM has been applied to predict equipment failure for decades and it has experienced a great evolution in methodologies. Conventionally, statistical approaches (e.g., regression-based methods[2–4] Gamma process-based methods [5–7] and Markovian-based methods [8–10]), have been widely employed to simulate the deteriorating patterns of machines. These approaches regard the machine health state as a random variable and use a history of condition monitoring data to build its density function. Despite the success that traditional statistical methods have achieved in certain applications, they are often inaccurate and incapable in modelling complex machinery components or systems. To face this challenge,

machine learning (ML) based methods, especially deep learning (DL), have gained momentum for automatically and quickly learning subtle patterns in big data and responding to specific needs with new methodological advancements. Considering the abundance of sensing data, advances in computational infrastructure, and ever-developing data analytical techniques, researchers have explored a variety of approaches to apply DL in PdM. Among all sorts of deep learning models, a recurrent neural network (RNN) is one of the most frequently used DL networks owing to its capability in dealing with time series data. As an advanced variant of RNN, long short term memory (LSTM) is broadly used in sequence prediction problems. Guo et al. [11] constructed an LSTM-RNN-based health indicator to predict remaining useful life (RUL) of bearings. Huang et al. [12] developed an LSTM-based RNN method for machine health prognostics under multiple operational conditions. Gugulothu et al. [13] used RNN to create an embedded structure that captures machine degradation patterns for time to failure (TTF) estimation.

Although RNN-based algorithms have been widely reported as a cutting-edge technique in PdM and have achieved great prediction accuracy in various applications, they are commonly recognized as "black box" approaches, due to their multi-layered and non-linear structure that consists of hundreds of thousands of parameters that must be determined through training [14]. These properties of an RNN make it nontransparent with its predictions difficult to trace. The consequence of this is that an RNN does not offer a comprehensive explanation on how decisions are made. What is needed is some method that allows for interpretation of the model and provides insight in terms of what the RNN has actually "learned." Such an interpretation would enable a better understanding on how the RNN associates the input features with its output decisions.

Interpreting neural networks is actually not a completely new concept in areas where image recognition and natural language processing are widely employed. To interpret neural networks, a few research studies have been published in these fields. Bohle et al. [15] successfully linked and explained patterns learned by a DL image classifier to the clinical magnetic resonance imaging (MRI)-based diagnosis for Alzheimers. Arras et al. [16] focused on explaining prediction in sentiment analysis that verifies whether a RNN classifier can (or cannot) detect important patterns in text datasets. In spite of the increasing interests in interpreting RNN models, little work exists in the PdM literature that involve RNN model interpretation approach in machine health analysis.

To fill this gap, we conducted analyses on an LSTM-RNN-based degradation diagnosis and prognosis model for monitoring manufacturing equipment performance. The model analyzed features extracted from vibration sensing data and predicts bearing health conditions. A layer-wise relevance propagation (LRP) technique was applied to decompose the model output relevance (also known as a relevance score) into contributions from each input neurons. During this process, the relevance was redistributed backwards from layer to layer and finally to the output layer, and the total amount of the relevance score in each layer is kept consistent. The relevance scores of the input layer are mapped into a 2D space that indicates both a time-series dimension and a feature-wise dimension respectively. The LRP results are analyzed on both dimensions with different labels to interpret individual classification decisions. Furthermore, the LRP results are also used to identify the most/least relevant features/time-steps input contributed to the RNN model. The obtained LRP results provide insights on how the data flows inside the model and demonstrate positive contribution/relevance to the neural network classification in the input space. The LRP has been proved to be effective in explaining neural network models and improving model accuracy and efficiency for PdM.

The remainder of the paper is organized as follows. Section 2 reviews the theory of the LSTM-RNN algorithm and its application in PdM. Section 3 describes the LRP technique for LSTM-RNN models. Section 4 illustrates a case study experiment where an RNN model for bearing health prediction is established and the LRP is implemented. Results and discussion of LRP analysis are shown in Section 5. Conclusions are drawn in Section 6.

2 Background on LSTM-RNN Models

Compared to general deep neural networks, the RNN is a specialized deep architecture for sequential modeling. Its variant LSTM was designed to further face the challenge of learning long-term dependencies [17], which has drawn increasing attention recently in many research areas. The following parts in this section review the basic theory of RNN and LSTM, as well as the related work in PdM using LSTM-RNN.

2.1 Recurrent Neural Network and Long-Short Term Memory Models

When dealing with the time sequential data generated from machine condition monitoring system, recurrent neural network is well known for its capability of capturing the dynamics of sequential data. Different from conventional neural network, neurons in RNN are strengthened by including edges that span adjacent time steps. These edges, called recurrent edges, augment the model data space with a dimension of time by forming cycles that are self-connected of a neuron to itself across time. For a simple recurrent network as shown in the left of Fig. 1(a), the dynamic of a neuron with recurrent edges can be described as:

$$\boldsymbol{h}^{(t)} = F(\boldsymbol{W}\boldsymbol{h}^{(t-1)} + \boldsymbol{U}\boldsymbol{x}^{(t)} + \boldsymbol{b}_{\boldsymbol{h}}), \tag{1}$$

$$\boldsymbol{y}^{(t)} = G(\boldsymbol{V}\boldsymbol{h}^{(t)} + \boldsymbol{b}_{\boldsymbol{y}}), \tag{2}$$

where, $\boldsymbol{h}^{(t)}$is the hidden layer activation at time t, $\boldsymbol{h}^{(t-1)}$ is the previous hidden representation, and $\boldsymbol{x}^{(t)}$ is the current input from the input layer. The RNN has input-to-hidden connections, hidden-to-hidden connections and hidden-to-output connections parametrized by the weight matrices $\boldsymbol{W}$, $\boldsymbol{U}$, $\boldsymbol{V}$, respectively. $\boldsymbol{b}_{\boldsymbol{h}}$ and $\boldsymbol{b}_{\boldsymbol{y}}$ are the bias parameters in hidden layer and output layer to allow offset learning. F and G are the activation functions for the two layers. $\boldsymbol{y}^{(t)}$ is the output of the RNN.

If we unfold the network in Fig.1(a) from left to right, it can be observed that the data propagation in the network is not cyclic but one-way along the time direction, which is similar to the propagation across layers. The difference lies in that the weights ($\boldsymbol{W}$) are shared across time steps. Thus, the network is trainable across multiple time steps using backpropagation algorithm. The algorithm introduced by Werbos [18] is called backpropagation through time (BPTT). However, since the weights across all time steps are always the same, the contribution of the input at time step t_1 to the time step t_2 will either move to infinity (explode) or decay to zero (vanish) as $t_2 - t_1$ grows large. The gradient of the loss will also explode or decay with respect to the input, with this behavior depending on whether $|\boldsymbol{W}| > 0$ or $|\boldsymbol{W}| < 0$ and on the activation function f.

In order to overcome the gradient exploding/vanishing problem, Hochreiter and Schmidhuber [19] introduced the long short-term memory model. The model has the architecture of a standard recurrent neural network with a hidden layer, but each neuron in the hidden layer is replaced by a memory cell structure (shown in Fig. 1(b)) with a core node called the state unit $\boldsymbol{s}^{(t)}$. Like a normal neuron in a hidden layer, the cell has external inputs from both the previous layer and previous state, and external outputs to the next time step and the following layer. But it also has an internal system of gating units that controls the flow of information via multiplication. For every time step t, the values of forget gate unit $\boldsymbol{f}_i^{(t)}$, state unit $\boldsymbol{s}^{(t)}$, input gate unit $\boldsymbol{g}_i^{(t)}$, output gate unit $\boldsymbol{q}_i^{(t)}$and output $\boldsymbol{h}_i^{(t)}$are updated from current input $\boldsymbol{x}_j^{(t)}$ and previous output $\boldsymbol{h}_j^{(t-1)}$. The computation procedure for an LSTM model at every step is provided below:

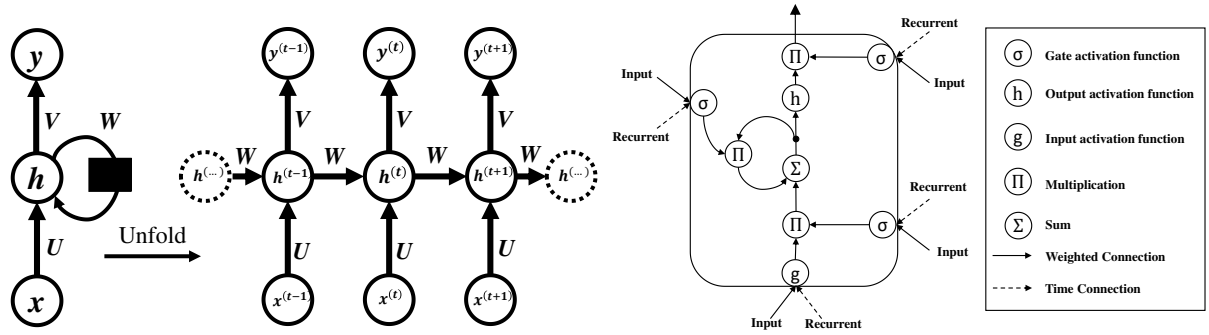
$$\boldsymbol{f}_i^{(t)} = \sigma\left(\boldsymbol{b}_i^f + \sum_j \boldsymbol{U}_{i,j}^f \boldsymbol{x}_j^{(t)} + \sum_j \boldsymbol{W}_{i,j}^f \boldsymbol{h}_j^{(t-1)}\right), \tag{3}$$

$$\boldsymbol{s}_i^{(t)} = \boldsymbol{f}_i^{(t)} \boldsymbol{s}_i^{(t-1)} + \boldsymbol{g}_i^{(t)} \sigma\left(\boldsymbol{b}_i + \sum_j \boldsymbol{U}_{i,j} \boldsymbol{x}_j^{(t)} + \sum_j \boldsymbol{W}_{i,j} \boldsymbol{h}_j^{(t-1)}\right), \tag{4}$$

$$\boldsymbol{g}_i^{(t)} = \sigma\left(\boldsymbol{b}_i^g + \sum_j \boldsymbol{U}_{i,j}^g \boldsymbol{x}_j^{(t)} + \sum_j \boldsymbol{W}_{i,j}^g \boldsymbol{h}_j^{(t-1)}\right), \tag{5}$$

$$\boldsymbol{q}_i^{(t)} = \sigma\left(\boldsymbol{b}_i^o + \sum_j \boldsymbol{U}_{i,j}^o \boldsymbol{x}_j^{(t)} + \sum_j \boldsymbol{W}_{i,j}^o \boldsymbol{h}_j^{(t-1)}\right), \tag{6}$$

$$\boldsymbol{h}_i^{(t)} = tanh\left(\boldsymbol{s}_i^{(t)}\right)\boldsymbol{q}_i^{(t)}. \tag{7}$$



(a) Recurrent Neural Network unfolded in times steps (b) Long Short Term Memory (LSTM) Cell

Fig. 1 Graphical Depictions of RNN Models

The three gate units and the state unit each have their own bias $\boldsymbol{b}_i$, input weights $\boldsymbol{U}_{i,j}$ and recurrent weights $\boldsymbol{W}_{i,j}$ and they are all activated by the sigmoid function $\sigma(\cdot)$. Finally, the hidden layer vector $\boldsymbol{h}_i^{(t)}$ is updated as the output of LSTM cell. In the forward direction, the LSTM can learn when and to what extent to let values in/out by activating an input/output gate. If both gates are closed, the value of hidden layer will neither grow nor shrink, so that no outputs or intermediate time steps will be affected. Similarly, the gradients can propagate back across many time steps stably without exploding or vanishing in backward propagation. In other words, gates are capable of learning when to let error in and when to let error out [17]. LSTM has been widely used for a variety of practical applications and has been shown to learn long-term dependencies more efficiently than the regular recurrent architectures.

2.2 Related work based on LSTM-RNN in PdM

The goal of a PdM system is to provide diagnostic and prognostic information on a monitored system (e.g., manufacturing equipment and turbine engine) from in-process sensory signals. Since these sensory signals are continuously being collected by the monitoring system, they naturally form a time series that could be used to estimate and predict the health condition of a manufacturing component (e.g. bearings and motors) or system (e.g. machine tools and turbine engines).

Let us assume that we periodically (perhaps once a day or once an hour) collect dynamic data from multiple sensors monitoring a system. Let $\boldsymbol{X_i} = \left[\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(T)}\right]$ be a dataset consisting of a series of consecutive observations for the *i*-th machine condition (e.g., the machine condition for the *i*-th day; for such a case, the monitoring interval between machine conditions is one day), where $\boldsymbol{x}^{(t)} \in \mathbb{R}^d$ is a vector of length *d* that contains data for multiple sensors sampled at time step *t*. There are in total *T* samples in the dataset. The corresponding machine health condition to $\boldsymbol{X_i}$ is denoted as y_i. LSTM-RNN model is used to predict $\overline{y_i}$ based on the sequential data $\boldsymbol{X_i}$.

Given the LSTM-RNN's strength in learning from sequential data, it has been used widely in many applications in PdM and achieved better performance compared to other methods. Dudukcu et al. [20] proposed a LSTM-based model combined with WaveNet to predict remaining useful life. A turbofan engine degradation simulation dataset was used to examine the model and close estimation to actual data was achieved. Zhang et al. [21] conducted an LSTM-based analysis to forecast equipment working condition. The method was evaluated for a cooling pump in a power station and the LSTM-based model had less RMS

error than that of autoregressive integrated moving average (ARIMA) model. Zhao et al. [22] evaluated LSTMs-based health monitoring systems for cutting tool wear. Basic and deep LSTMs were used to predict the tool wear with sensory data and a deep LSTM was able to outperform several state-of-art baseline methods. Research relating to mechanical equipment anomaly detection was completed by Li et al. [23]. They developed a LSTM-based classifier model for detecting abnormal condition of rotating equipment and realized 99% accuracy in an unsupervised learning environment.

As is evident, many investigators have applied LSTM-RNN to PdM with great success. They have not, however, examined the LSTM-RNN evolves in response to newly acquired input data. The next section will introduce LRP as a tool to explain how the LSTM-RNN method "learns" from data.

3 LRP Analysis for LSTM-RNN models

3.1 General LRP theory

Given a trained neural network classifier, one of the things we are interested in knowing is how much each input contributes to a target class of interest, *c*, or, a relevance score of each input with respect to *c*. Layer-wise relevance propagation, proposed by Binder and Bach et al. [24] in 2015, is such an explanation technique to compute relevance. The main idea of LRP is to assign a relevance score to each individual input by tracing back their contribution to the final prediction, $f(x)$, layer by layer. The LRP process follows a conservation principle that the total amount of relevance distributed to one layer should be equal to the total amount of relevance distributed in the previous layer. Say that *m* and *n* are two consecutive layers of a neural network, the relevance scores satisfy the following rule:

$$\sum_i R_i^{(m)} = \sum_i R_i^{(n)} = f(x), \tag{8}$$

where, $R_i^{(m)}$and $R_i^{(n)}$ are the relevance scores of individual neurons in layers *m* and *n* respectively. In order to fit the characteristics of different neural network structure, there are various rules defining how the relevance scores propagate between two layers in compliance with Eq. (8). A basic rule, namely LRP-0, is shown in Eq. (9):

$$R_i^{(m)} = \sum_j \frac{z_{i,j}}{\sum_k z_{k,j}} R_j^{(n)}, \tag{9}$$

where, $z_{i,j}$ is the contribution/relevance received by neuron *j* in layer *n* from activated neuron *i* in layer *m*; $\sum_k z_{k,j}$ is the total contribution/relevance sent to neuron *j* from all connected neurons in layer *m* before a non-linear activation function is applied. The conservation principle is evident in this equation; it also applies to situations such as zero weight, deactivation, and disconnected neurons.

Although LRP-0 rule has many appealing properties, when applied to real cases, robustness needs to be considered as well as other enhancements. In this research, two enhanced LRP rules were used:

1) Epsilon rule (LRP-ε):

$$R_i^{(m)} = \sum_j \frac{z_{i,j}}{\varepsilon+\sum_k z_{k,j}} R_j^{(n)}. \tag{10}$$

Compared to the basic rule, the denominator includes a small positive term ε to ensure numerical stability.

2)Alpha-beta rule (LRP-αβ):

$$R_i^{(m)} = \sum_j \left(\alpha \frac{{z_{i,j}}^+}{\sum_k {z_{k,j}}^+} - \beta \frac{{z_{i,j}}^-}{\sum_k {z_{k,j}}^-} \right) R_j^{(n)}, \tag{11}$$

where, ${z_{i,j}}^+$ and ${z_{i,j}}^-$ refer to the positive and negative contributions from the higher layer *n*, respectively, and α and β are two parameters to control the weights of positive contributions and negative contributions. In accordance with the conservation principle, $\alpha + \beta$ should equal to one. The rule is designed to favor the

effect of positive contributions over the negative ones, so that certain stability and interpretation could be brought to the final results. By choosing the value of coefficient α and β carefully, one can manually control the importance of positive and negative contributions.

3.2 Apply LRP for LSTM

Besides linear mapping computation in multilayer perceptron structures, RNN-LSTM as well as other gated neural networks have a special computation known as multiplicative interaction. In such computation, there are two neurons multiplied by each other: one serves as signal while the other serves as a gate, which controls how much the signal influences the output:

$$a_p = f(z_g) \cdot g(z_s) , \tag{12}$$

where, $f(\cdot)$ is the activation functions for the gate unit and $g(\cdot)$ is the activation function for the signal unit, z_g and z_s are two neuron values sent to gate and signal unit from previous layers, respectively.

Unlike linear mapping, the nonlinearity lying in multiplicative interaction has its own inherent difficulties associated with redistributing relevance to the previous layer. In such a case where an activation is gained by the value of a gate neuron multiplying the value of a signal neuron, one widely accepted redistribution strategy [25] is called "signal-take-all", which refers to:

$$(R_g, R_s) = (0, R_p) , \tag{13}$$

where, R_g and R_s are the relevance scores assigned to the gate neuron and the signal neuron. The signal neuron takes all the relevance R_p from the upper layer while the gate neuron takes zero to obey the conservation principle. One way of interpreting this strategy is that the gate controls the flow of information, but is not information itself. Rather, information is wholly embedded within the signal. Although it seems that the contribution of z_g is totally neglected, the effect of z_g has been actually considered during the process of computing the value of R_p from upper-layered structure.

Another particular operation in LSTM is the accumulation as stated in Eq. (4). Controlled by the forget gate, the new state unit is updated from the previous one. The corresponding LRP rule is similar to the one for multiplicative interaction. However, while redistributing relevance scores to the previous time step input, the value tends to decrease exponentially backwards with the time step.

4 Case Study Experiment

It is now desired to examine the effectiveness of using LRP techniques to interpret LSTM-RNN models, To do this, a model developed by Wu et al. [26] based on data published by the Center for Intelligent Maintenance Systems (IMS) at the University of Cincinnati [27] was used as a case study. The model takes time-series data of multiple features extracted from vibration signal as the input, and outputs classification results in terms of 4 categories/labels as an estimate of the health state of the bearing.

4.1 Experiment setup and data preprocessing

The data for this experiment [27] was from a setup that had four Rexnord ZA-2115 double row bearings that were installed on a shaft. The shaft was driven by an AC motor via friction belts with a constant rotational speed of 2000 RPM. A radial load of 6000 lbs. (2722 kg) was applied to the shaft and bearings by a spring mechanism. Accelerometers (PCB model #353B33) were installed on the housing of each of the four bearings. A sketch of the test rig including the sensor placement is shown in Fig. 2. Run to failure experiments on bearing health degradation were conducted, and 12 sets of accelerometer data were collected. For each set of accelerometer data, a 1-second vibration signal time series snapshot was collected every 10 minutes. Each snapshot consisted of 20,480 data points (i.e., the accelerometer signal was sampled at a rate of 20 kHz). A NI DAQ Card 6062E was used in data collection process.

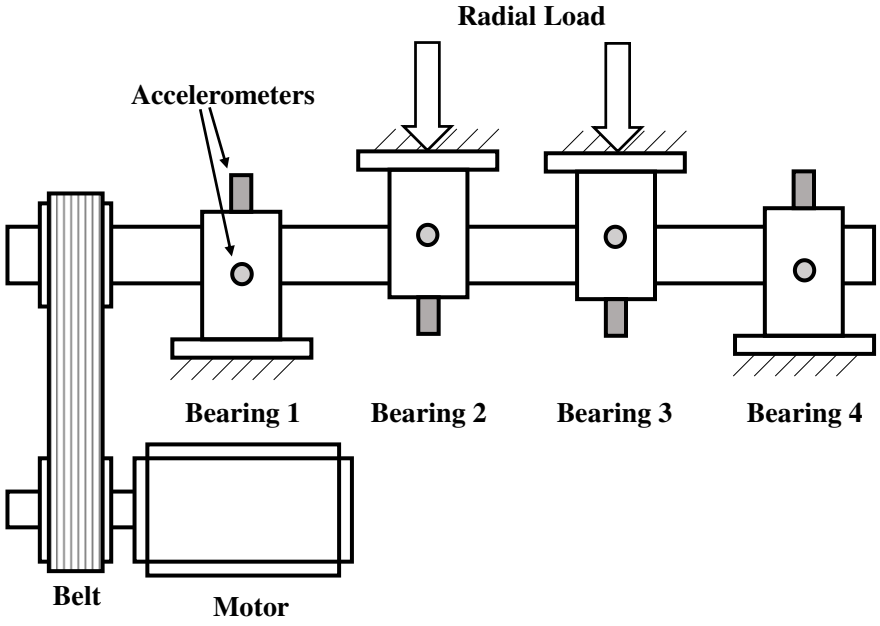


Fig. 2 IMS bearing test rig and sensor placement [28]

Through the life of the bearings, each vibration snapshot was analyzed in the time and frequency domains, and 35 features were extracted and normalized as a feature set. The names of the features and corresponding domains are listed in the Appendix as Table A1. The listed features were selected from the most commonly used features in the bearing health diagnosis and prognosis research literature [29]. In order to provide the model with time-series information for learning, a series of consecutive feature sets were utilized (one feature set every 10 minutes). A time sequence window was chosen to be 15 in this case, thus a 15×35 matrix was formed as the input to the model. This time sequence window moved along the feature sets series from the first 15 feature sets. Every time the window moved towards the end of bearing life by one time step, a new matrix will be generated as one input matrix. For example, a feature sets series with length of 2000 can generate 1986 matrices based on this "moving window". The health conditions of the bearings were classified into 4 states (labeled as "Normal I", "Normal II", "Warning", and "Failing") in terms of the TTF of 100%-50%, 50%-20%, 20%-5%, 5%-0%. For each input matrix, the corresponding label/output is the health state at the last time step for that window (time step 15).

4.2 LSTM-RNN Architecture

In this case study, a multi-layered deep neural network architecture was designed to classify the health states of bearings. The LSTM-RNN model with specific layer details is shown in Table 1. The input layer takes a 15×35 matrix and sends it into the LSTM layer. As the core layer in this network, the LSTM layer has 30 neurons with a hyperbolic tangent (tanh) activation function. Each neuron in this layer spans 15 time-steps in accordance with the input time dimension. The LSTM layer is connected to a fully connected (dense) layer with 50 neurons (Dense #1) and an activation function of a rectified linear unit (ReLU), followed by two dense layers with 10 (Dense #2) and 4 neurons (Dense #3), respectively, both activated by a ReLU function. The last layer is the output layer of a softmax function that takes the maximum output from the previous layer (Dense#3) as the classification result. The selection of hyper parameters, such as

neuron numbers in each layer, is based on existing successful LSTM-RNN structure references. This design was based on the idea that the network should first expand the data space from the input to generate abundant information, then compress the space to automatically select necessary information, and finally make a decision via a single output.

Table 1. Layer details of LSTM-RNN

Layer	Output size	Activation function
Input	15×35	tanh
LSTM	30	ReLU
Dense#1	50	ReLU
Dense#2	10	ReLU
Dense#3	4	ReLU
Output	1	softmax

The LSTM-RNN model was constructed using Keras API for Tensorflow and python coding. During the training phase, 10 out of 12 sets of experiment data were used to train the model. The mean squared error (MSE) was used as the loss function and adaptive moment estimation (Adam) was selected as the optimizer. After 100 epochs of iterations, the model reached an accuracy above 99% and a loss below 0.001. Then, the other two sets of experiment data were imported into the trained model to evaluate its adequacy. From the testing phase, an over-all accuracy of 90.07% was observed. It should be noted that the classification accuracy in terms of "Warning" and "Failure" states was over 95%. The results from the testing phase are shown in Fig. 3 as a confusion matrix.

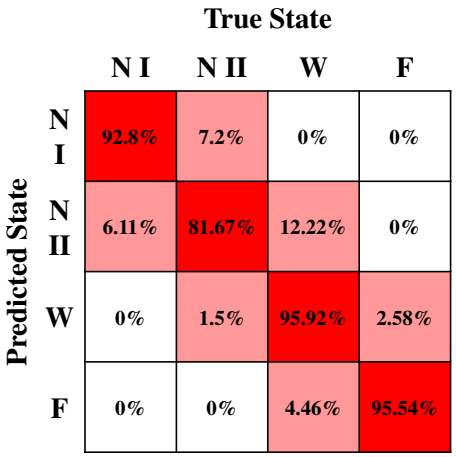


Fig. 3 Results from the testing phase of the LSTM-RNN model

4.3 Applying LRP to the LSTM-RNN model

As noted above, the LSTM-RNN model was evaluated using two of the twelve datasets. For each of these evaluations, the input matrix from the dataset was processed by the LSTM-RNN model to estimate the

bearing health state. The LRP technique was then used in concert with the structure of the LSTM-RNN model, health states, and input matrices to analyze the degradation patterns and distinguishable information from different categories. In these analyses, three aspects were of particular interest: (1) how the inputs, time step-wise and feature-wise, contribute to the classification decision, (2) how the features contribute to the output through different degradation stages, and (3) how LRP analysis could improve the accuracy and efficiency of the LSTM-RNN model.

The LRP-αβ rule was employed to reduce the effect of negative contributions (relative to positive contributions), and α and β were set to 2 and -1. For each sample, the relevance score of the output layer neuron was set to 1 for consistency. While the relevance score back-propagated to the Dense#3 layer where values of 4 neurons representing 4 classes were compared, only the neuron of the predicted class received a whole relevance score of 1 (the other three received 0). The rest of relevance propagation towards the input layer would follow the principles introduced in Section 3.1. Finally, for each sample, a new 15×35 matrix was generated with relevance scores mapping to the input space. A flowchart is shown in Fig. 4 to demonstrate the whole process.

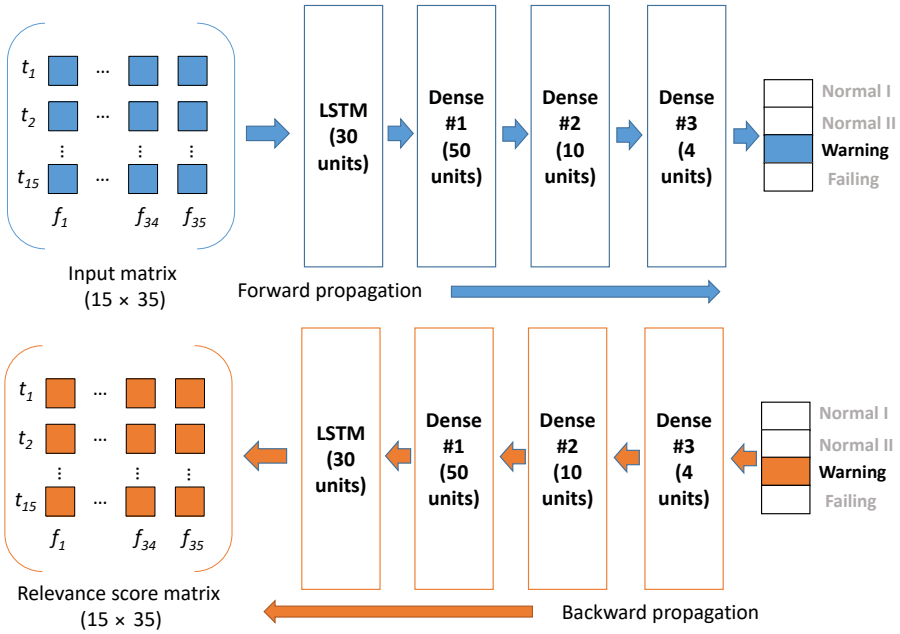


Fig. 4 Flowchart for interpreting LSTM-RNN via LRP

5 LRP Results and Discussion

5.1 General LRP results in features and time steps

An LRP analysis provides relevance scores that are assigned to the input space (in this case a 2D space). Thus, a 2D relevance score matrix will be created for each input dataset. LRP provides insight into how the LSTM-RNN model uses contributions from the input layer to make a decision. Of course, it is necessary to first scrutinize the LRP technique to verify its ability to interpret the model accurately. Also, in many cases, to visualize a relevance matrix, a heat map is utilized. For such a heat map, the relevance scores are converted to colors to create a map. However, unlike LRP analysis for an image recognition application, where the heat map overlaid on an image gives an intuitive explanation on the patterns learned by the model, a little bit more care must be exercised in interpreting a heat map overlaid on a time-series feature matrix in this case.

As is evident from Fig. 4, the vertical dimension in the relevance score matrix is associated with time steps, and the horizontal dimension is associated with features. If it was desired to know which time steps are most relevant, the matrix could be collapsed in the feature dimension by summing the relevance scores across features to form a column vector. The key time steps could then easily be discerned. Likewise, by summing the relevance scores across time steps, a row vector could be gained to know the key features. After doing the time steps-dimensional summation for all the correctly classified samples, a distribution of features is shown in Fig. 5 (the features are shown in Table A1. in the Appendix). This summed relevance score illustrates the impact of the features on the output of the LSTM-RNN model. It is noted that there are significant variations in the scores associated with the features. Some features (such as #4, 10, 20, and 21) have large sums, indicating a strong impact on the output, while others (such as # 2, 3, 16, and 28) have small sums, which suggests little importance in terms of the output.

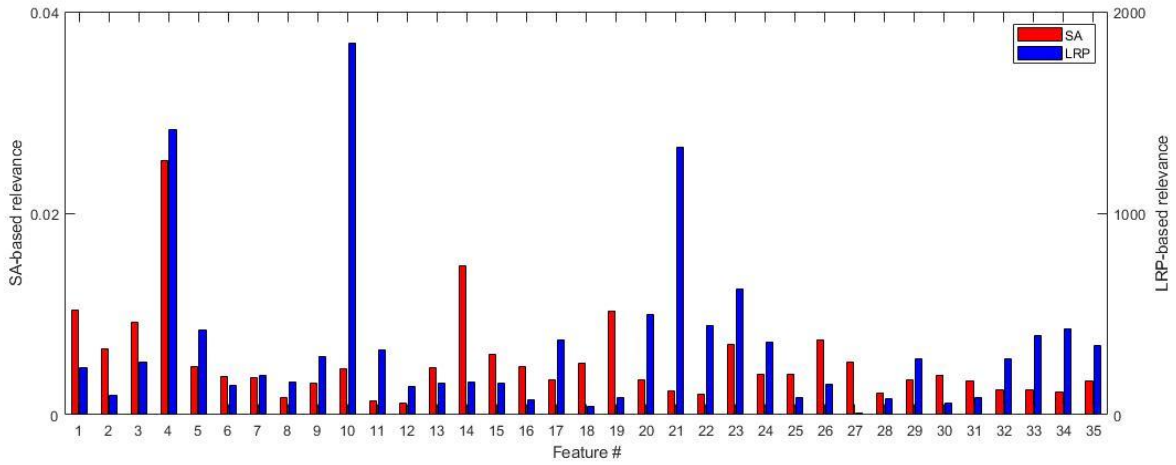


Fig. 5 Summed relevance score for features based on LRP and SA

In order to validate LRP's capability of interpreting LSTM-RNN model, a comparison was made with a commonly used approach for examining the importance of variables within a nonlinear model, namely sensitivity analysis (SA) [30]. SA evaluates relevance based on the model's locally evaluated gradient or some other local measure of variation. A common formulation of sensitivity analysis based on gradients defines relevance scores as:

$$\mathrm{R}_i(\boldsymbol{x}) = \left(\frac{\partial f(\boldsymbol{x})}{\partial x_i}\right)^2 , \tag{14}$$

where, the gradient is evaluated at the data point $\boldsymbol{x}$. The summed relevance scores of features based on this SA approach is also shown in Fig. 5 for comparison. It presents a different profile, i.e., different set of important features, from the LRP method. To quantitatively validate that LRP provides a better interpretation, feature removal experiments were performed. The idea of these experiments was that the removal of the features with the most relevance to the model output would most likely change the model decision. In these experiments, the LSTM-RNN model was retrained with the same number of important features removed from the input based on either the LRP and SA approach. The losses in testing accuracy were compared for the two approaches. Four experimental trials were conducted (Trial 1: the most relevant feature was removed, Trial 2: the two most relevant features removed, and so forth) for each approach (LRP vs. SA). For each trial, 5 sets of initializing parameters were considered during the training phase to obtain 5 models; the same parameter sets were used for each approach (LRP and SA) for a given trial. It should be noted that the parameters changed from trial to trial because of changes in the dimensionality of the input

matrix. For each trial, the model was initialized with the same parameters for both approaches (but, of course, the input matrix was different from trial to trial) and an average accuracy loss was calculated from 5 trainings (corresponding to 5 sets of initial parameters). The features removed in the experiments are listed in Table 2. Since feature #4 is the most important features in SA and second most important feature in LRP, the removal of feature #4 is not considered to avoid potential confusion. The results are shown in Fig. 6. It is clearly seen that the accuracy losses caused by the removal of relevant features based on LRP were much more significant than that of SA. This suggests that the features identified by LRP are more important to the accuracy of the LSTM-RNN model than features identified with SA.

Table 2. Features removed in each experimental trial

Trial	LRP	SA
1	#10	#14
2	#10,21	#14,1
3	#10,21,20	#14,1,19
4	#10,21,20,23	#14,1,19,26

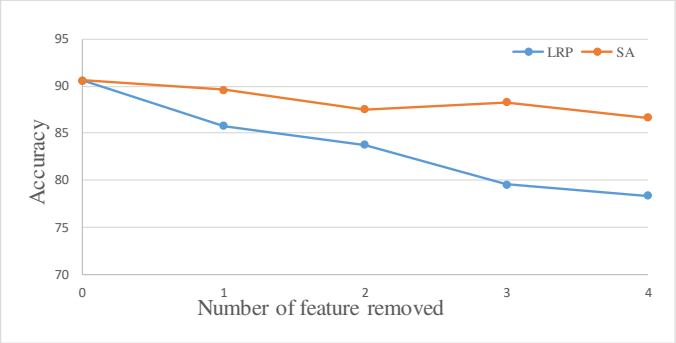


Fig. 6 Accuracy loss by removing features according to LRP and SA

In an analogous manner to what was done by summing across time steps for each feature, a summation was performed across features to show the summed relevance scores for time steps as shown in Fig. 7. The same relevance scores were also computed using SA for comparison. While the SA-based relevance score does not show a distinct pattern related to time, it is clearly seen that the LRP-based relevance score grows with time. This result indicates that LRP is suggesting that the LSTM-RNN model assigns more weights to recent information than past information. The LRP result also matches the health state label that corresponds to the last time step in the input. The most recent assessment of health condition is obviously very relevant.

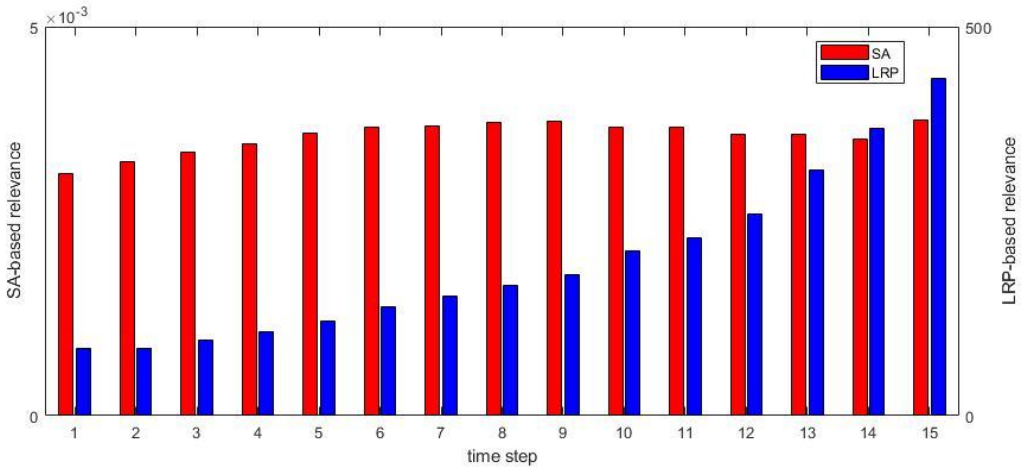


Fig. 7 Relevance distribution of time steps

5.2 LRP results regarding different categories

The results in Section 5.1 demonstrated the extent to which features and time steps contribute to the decisions made by the model. However, this discussion did not examine how the important features, from a relevance standpoint, may be different depending on the health state (category). To gain insights on how features could influence an individual classification decision, relevance scores of features for each category were computed separately. The relevance scores were normalized to eliminate the effect of different sample sizes for the categories. Figure 8 shows the relevance scores of features for the 4 health states ("Normal I", "Normal II", "Warning", and "Failing"). It is seen that the impact of certain features on one state could be dramatically different from that on another state. For example, feature #4 contributes more to the classification of "Failing" state compared with the other three states, while feature #10 has major relevance to the identification of the "Normal II" state, but has little relevance to the "Failing" state. Such observations suggest that a feature could strongly influence being identified in one health state while playing a very small role with being classified in another state. Seemingly, when the model is identifying different bearing health states, the features are weighted differently by the LSTM-RNN model. In other words, the relevance of features to the model output varies by category.

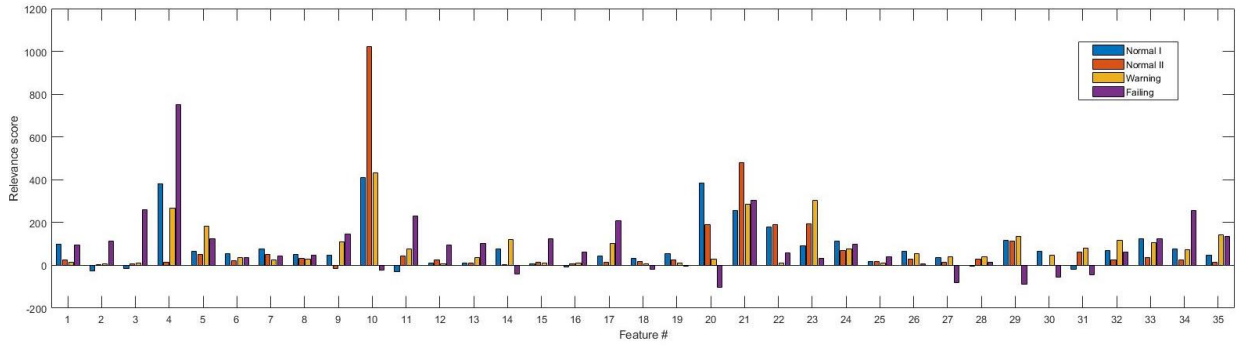


Fig. 8 Relevance distribution of features with different classes

To validate these findings, experiments were conducted by retraining the model with a single feature removed and examining how this influences the classification accuracy of each state. In PdM, the major focus is on identifying when a critical state is entered ("Warning" and "Failing") for safety and maintenance purposes. In this experiment, feature #4 (Entropy) was removed from the input since it has the most

relevance to the "Failing" state. A second set of input data (serving as a control case) was constructed by removing a relatively unimportant feature (feature #2, Variance). The method was applied to both data inputs to obtain LSTM-RNN models. These two models were trained and tested in the same manner as the original model but with one less feature input (the input is a 14×35 matrix). Confusion matrices of the testing results from the two retrained models, as well as the original model, are presented in Fig. 9.

Fig. 9 Confusion matrices for the three models

Compared to the original model, the removal of feature #4 (Entropy) resulted in accuracy declines in the identification of "Warning" and "Failing" state of 13.7% (95.9% to 82.2%) and 15.3% (95.6% to 80.3%) respectively. It aggravated the confusion between those two states as well. In comparison, the removal of feature #2 did not cause prominent changes in the accuracy of true state prediction, which corresponds to its slight relevance from the LRP analysis.

This result might be explained when we take a close look at the two features discussed above. According to the definitions in [31], variance simply measures the dispersion of a signal about its mean value, while entropy is a calculation of the uncertainty and randomness in the information content in the data, which may better reflect the dynamics of bearing vibration. In Fig. 10, entropy and variance obtained from a raw vibration signal are plotted throughout the lifespan of a bearing. As is evident, there is merely a sudden surge in the variance just before the bearing fails; virtually no warning is provided. By comparison, entropy starts increasing when the bearing approaches the later stage of its lifespan; a relatively early warning could be provided to the model to identify the critical states. Again, this could also be regarded as evidence that LRP's interpretation reveals the patterns behind the data.

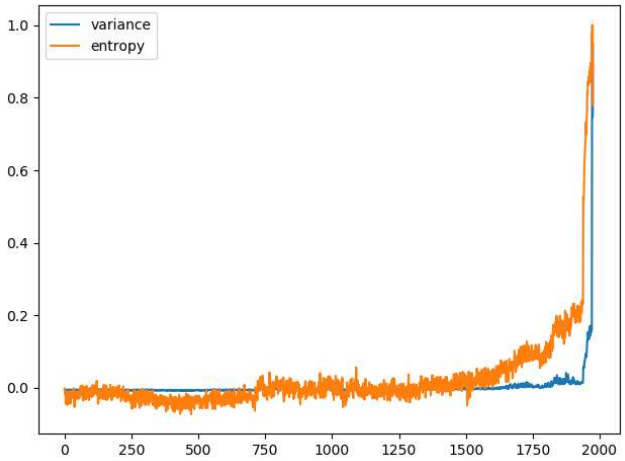


Fig. 10 Entropy and variance during the lifespan of a bearing

5.3 Improving the model efficiency and accuracy with LRP

The analysis above has clearly shown how LRP may be used to explain the relevance between the input and output for an LSTM-RNN model. The reflected relevance has been validated by several experiments. By analyzing those relevance scores of input layer and the middle layer neurons, several observations were made. According to these observations, some methods designed for model efficiency and accuracy improvement are discussed below.

When computing the relevance score of each feature to the model output, some features were found to have very limited impact on the model decision making. When checking the relevance scores regarding each of the four categories, the relevance scores of certain features remain low, which suggests that those features may not substantially contribute to model decision making. This observation leads to a promising approach to reduce model complexity, by removing insignificant features so as to reduce model input size, while achieving comparable performance to the original model. Iterative trials of retraining the model were performed. These trials removed different combinations of features while keeping the same model architecture, except for the input layer (the number of input layer neurons should be the same as the number of input features). Finally, a new model with 23 input features (12 features removed: #2, 3, 6, 12, 13, 16, 18, 19, 25, 28, 30, 34) achieved an acceptable overall testing accuracy of 87.79% compared to the original 90.6%. This model also retained good performance in critical states identification: 90.34% and 100% for "Warning" and "Failing" states respectively.

The relevance scores of the neurons in the middle fully-connected layers (Dense#1, Dense#2) were computed. It was found that around 40% (Dense #1) and 20% (Dense #2) of the neurons in these layers had relevance scores equal or nearly equal to 0. This means that there is redundancy in the original model and fewer neurons could be used. Therefore, we shrank the number of neurons in those layers by 40% in Dense #1 and 20% in Dense #2 respectively. By making this adjustment the total number of trainable parameters was reduced from 9,904 to 5,394. After training the reduced parameter model with the same training procedure as the original model, an overall test accuracy of 89.81% (compared to the original 90.6%) was achieved and the classification performance was comparable to the original model.

When checking the individual sample heat maps that were correctly and wrongly classified, it may be seen that there is a large number of pixels with excessive relevance scores (the yellow pixels in Fig. 11 that exceed 0.10) for the wrongly classified sample heat maps. Based on this observation, a double-check mechanism was established, which could be used to evaluate whether a prediction made by the model is trustworthy. We set the threshold as: when the number of pixel values exceeding 0.1 is greater than 3, the prediction made by the model should be regarded as questionable. For a test dataset containing 2142 samples, the model predicted 2035 samples with the correct classes and 107 samples wrong. With this mechanism applied, more than 80% of wrongly predicted samples were found to be questionable. Thus, this mechanism based on LRP analysis was proved to be an effective way to further improve model accuracy.

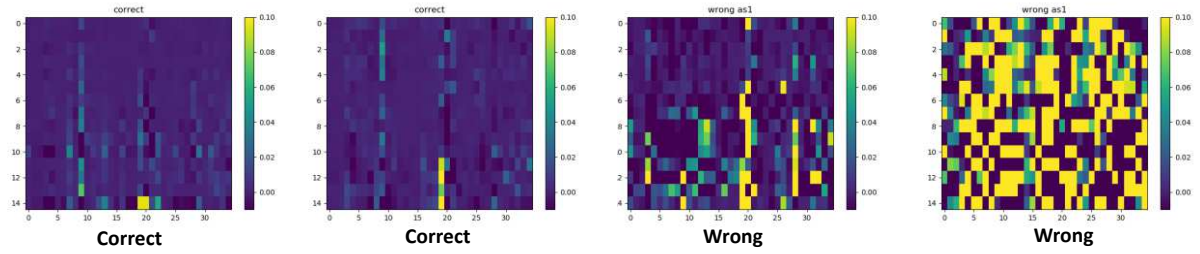


Fig. 11 Heat maps of two correctly classified and two wrongly classified samples

6 Conclusion

Many research efforts have been carried out to apply recurrent neural network models in the field of predictive maintenance. However, there have been few studies focused on interpreting such models and understand their decision making process. This paper addressed this gap by employing model explanation techniques to investigate the contributing factors to the decision made by a LSTM-RNN model for bearing health condition estimation.

The LRP technique was used to inspect the relevance distribution in the input space. Analysis was conducted on the feature contributions and time steps contributions. A comparison was made with a common approach i.e., sensitivity analysis, to understanding the role of factors in a model, and it demonstrated LRP's superior capability for model interpretation. Further analysis into individual classification categories showed that the LSTM-RNN model adopted information from the input data differently when making individual decisions. The LRP analysis also proved to be useful for model efficiency improvement by removing irrelevant features from the input and optimizing model architecture so as to have fewer parameters. The potential for LRP to help achieve higher classification accuracy was also illustrated.

Overall, these results demonstrated that LRP has great potential for improving neural network transparency and changing the "black box" impression of RNN. It is also promising to see LRP being applied to a predictive maintenance scenario. It is expected that additional maintenance challenges will be addressed in future research.

Acknowledgments

The authors gratefully acknowledge the support of the Wabash Heartland Innovation Network (WHIN) grant at Purdue University.

Declarations

Funding This research was supported by the Wabash Heartland Innovation Network (WHIN) grant at Purdue University.

Conflicts of Interest/Competing interests The authors declare no conflicts of interest or competing interest.

Availability of data and material The raw data used in case study was from NASA Ames Prognostics Data Repository (http://ti.arc.nasa.gov/project/prognostic-data-repository).

Authors' contributions

Haiyue Wu: Conceptualization, investigation, methodology, writing-original draft.

Aihua Huang: Writing-reviewing and editing.

John W. Sutherland: Supervision, writing-reviewing and editing.

Ethical approval The authors would like to declare that the work described was original which has not been published previously or not under consideration for publication elsewhere.

Consent to participate Not applicable

Consent to publish All of the authors have reviewed the final version of the manuscript and approve it for publication.

Reference

1. Monostori L, Kádár B, Bauernhansl T, Kondoh S, Kumara S, Reinhart G, Sauer O, Schuh G, Sihn W, Ueda K (2016) Cyber-physical systems in manufacturing. CIRP Ann 65:621–641. https://doi.org/10.1016/j.cirp.2016.06.005
2. Menard S (2011) Applied Logistic Regression Analysis. Appl Logist Regres Anal. https://doi.org/10.4135/9781412983433
3. Wang G, Luo Z, Qin X, et al (2008) Fault identification and classification of rolling element bearing based on time-varying autoregressive spectrum. Mech Syst Signal Process 22:934–947. https://doi.org/10.1016/j.ymssp.2007.10.008
4. Salem O, Guerassimov A, Mehaoua A, et al (2014) Anomaly Detection in Medical Wireless Sensor Networks using SVM and Linear Regression Models. Int J E-Health Med Commun 5:20–45. https://doi.org/10.4018/ijehmc.2014010102
5. Yan HC, Zhou JH, Pang CK (2015) Gamma process with recursive MLE for wear PDF prediction in precognitive maintenance under aperiodic monitoring. Mechatronics 31:68–77. https://doi.org/10.1016/j.mechatronics.2015.05.009
6. Xu W, Wang W (2012) An adaptive gamma process based model for residual useful life prediction. Proc IEEE 2012 Progn Syst Heal Manag Conf PHM-2012 3–6. https://doi.org/10.1109/PHM.2012.6228785
7. Wei Q, Xu D (2014) Remaining useful life estimation based on gamma process considered with measurement error. ICRMS 2014 - Proc 2014 10th Int Conf Reliab Maintainab Saf More Reliab Prod More Secur Life 645–649. https://doi.org/10.1109/ICRMS.2014.7107275
8. Zhou D, Yu Z, Zhang H, Weng S (2016) A novel grey prognostic model based on Markov process and grey incidence analysis for energy conversion equipment degradation. Energy 109:420–429. https://doi.org/10.1016/j.energy.2016.05.008
9. Ertunc HM, Loparo KA, Ocak H (2001) Tool wear condition monitoring in drilling operations using hidden Markov models (HMMs). Int J Mach Tools Manuf 41:1363–1384. https://doi.org/10.1016/S0890-6955(00)00112-7
10. Ocak H, Loparo KA, Discenzo FM (2007) Online tracking of bearing wear using wavelet packet decomposition and probabilistic modeling: A method for bearing prognostics. J Sound Vib 302:951–961. https://doi.org/10.1016/j.jsv.2007.01.001
11. Guo L (2017) A recurrent neural network based health indicator for remaining useful life prediction of bearings. Bol Tec Bull 55:585–590. https://doi.org/10.1016/j.neucom.2017.02.045
12. Huang CG, Huang HZ, Li YF (2019) A Bidirectional LSTM Prognostics Method Under Multiple Operational Conditions. IEEE Trans Ind Electron 66:8792–8802. https://doi.org/10.1109/TIE.2019.2891463
13. Gugulothu N (2017) Predicting Remaining Useful Life using Time Series Embeddings based on Recurrent Neural Networks. CEUR Workshop Proc 2657:1–9. https://doi.org/10.1145/nnnnnnn.nnnnnnn
14. Montavon G, Samek W, Müller KR (2018) Methods for interpreting and understanding deep neural networks. Digit Signal Process A Rev J 73:1–15. https://doi.org/10.1016/j.dsp.2017.10.011
15. Böhle M, Eitel F, Weygandt M, Ritter K (2019) Layer-wise relevance propagation for explaining

deep neural network decisions in MRI-based Alzheimer's disease classification. Front Aging Neurosci 10:. https://doi.org/10.3389/fnagi.2019.00194

16. Arras L, Montavon G, Müller K-R, Samek W (2018) Explaining Recurrent Neural Network Predictions in Sentiment Analysis. 159–168. https://doi.org/10.18653/v1/w17-5221

17. Hochreiter S The vanishing gradient problem during learning recurrent neural nets and problem solutions

18. Werbos PJ (1990) Backpropagation Through Time: What It Does and How to Do It. Proc IEEE 78:1550–1560. https://doi.org/10.1109/5.58337

19. Hochreiter S (1997) Long Short-Term Memory. Neural Computation 9:1735–1780

20. Dudukcu HV, Taskiran M, Kahraman N (2020) LSTM and WaveNet Implementation for Predictive Maintenance of Turbofan Engines. 20th IEEE Int Symp Comput Intell Informatics, CINTI 2020 - Proc 151–156. https://doi.org/10.1109/CINTI51262.2020.9305820

21. Zhang W, Guo W, Liu X, et al (2018) LSTM-Based Analysis of Industrial IoT Equipment. IEEE Access 6:23551–23560. https://doi.org/10.1109/ACCESS.2018.2825538

22. Zhao R, Wang J, Yan R, Mao K (2016) Machine health monitoring with LSTM networks. Proc Int Conf Sens Technol ICST 17–22. https://doi.org/10.1109/ICSensT.2016.7796266

23. Li Z, Li J, Wang Y, Wang K (2019) A deep learning approach for anomaly detection based on SAE and LSTM in mechanical equipment. Int J Adv Manuf Technol 103:499–510. https://doi.org/10.1007/s00170-019-03557-w

24. Binder A, Bach S, Montavon G, et al (2016) Layer-wise relevance propagation for deep neural network architectures. Lect Notes Electr Eng 376:913–922. https://doi.org/10.1007/978-981-10-0557-2_87

25. Arras L, Arjona-Medina J, Widrich M, et al (2019) Explaining and Interpreting LSTMs. Lect Notes Comput Sci (including Subser Lect Notes Artif Intell Lect Notes Bioinformatics) 11700 LNCS:211–238. https://doi.org/10.1007/978-3-030-28954-6_11

26. Wu H, Huang A, Sutherland JW (2020) Avoiding Environmental Consequences of Equipment Failure via an LSTM-Based Model for Predictive Maintenance. Procedia Manuf 43:666–673. https://doi.org/10.1016/j.promfg.2020.02.131

27. University of Cincinnati Bearing Data Set, NASA Ames Prognostics Data Repository http://ti.arc.nasa.gov/project/prognostic-data-repository, NASA Ames Research Center, 2006

28. Qiu H, Lee J, Lin J, Yu G (2006) Wavelet filter-based weak signature detection method and its application on rolling element bearing prognostics. J Sound Vib 289:1066–1090. https://doi.org/10.1016/j.jsv.2005.03.007

29. Lee J, Wu F, Zhao W, et al (2014) Prognostics and health management design for rotary machinery systems - Reviews, methodology and applications. Mech Syst Signal Process 42:314–334. https://doi.org/10.1016/j.ymssp.2013.06.004

30. Zurada JM, Malinowski A, Cloete I (1994) Sensitivity analysis for minimization of input data dimension for feedforward neural network. Proc - IEEE Int Symp Circuits Syst 6:447–450. https://doi.org/10.1109/iscas.1994.409622

31. Caesarendra W, Tjahjowidodo T (2017) A Review of Feature Extraction Methods in Vibration-Based Condition Monitoring and Its Application for Degradation Trend Estimation of Low-Speed Slew Bearing. Machines 5:21. https://doi.org/10.3390/machines5040021

Appendix

Table A1. Features used in LSTM-RNN Model (Feature details could be found in [11] and [31])

No.	Name	Domain
Fea#1	Mean	Time domain
Fea#2	Variance	Time domain
Fea#3	RMS	Time domain
Fea#4	Entropy	Time domain
Fea#5	Skewness	Time domain
Fea#6	Kurtosis	Time domain
Fea#7	Shape factor	Time domain
Fea#8	Crest factor	Time domain
Fea#9	Upper bound of Histogram	Time domain
Fea#10	Lower bound of Histogram	Time domain
Fea#11	Impulse factor	Time domain
Fea#12	Margin factor	Time domain
Fea#13	Mean frequency	Time domain
Fea#14	Frequency Centre	Frequency domain
Fea#15	Root mean square frequency	Frequency domain
Fea#16	Standard deviation frequency	Frequency domain
Fea#17	Peak to peak	Time domain
Fea#18	Spectral Skewness	Frequency domain
Fea#19	Spectral Kurtosis	Frequency domain
Fea#20	Spectral Peak Ratio(outer)	Frequency domain
Fea#21	Spectral Peak Ratio(inner)	Frequency domain
Fea#22	Spectral Peak Ratio(rolling element)	Frequency domain
Fea#23	Related Similarity 2	Frequency domain
Fea#24	Related Similarity 3	Frequency domain
Fea#25	Related Similarity 4	Frequency domain
Fea#26	Related Similarity 5	Frequency domain
Fea#27	Related Similarity 6	Frequency domain
Fea#28	Wavelet Energy ratio 1 (Energy ratios of eight frequency sub-bands from haar wavelet package transform)	Time and frequency domain
Fea#29	Wavelet Energy ratio 2	Time and frequency domain
Fea#30	Wavelet Energy ratio 3	Time and frequency domain
Fea#31	Wavelet Energy ratio 4	Time and frequency domain
Fea#32	Wavelet Energy ratio 5	Time and frequency domain
Fea#33	Wavelet Energy ratio 6	Time and frequency domain
Fea#34	Wavelet Energy ratio 7	Time and frequency domain

Fea#35	Wavelet Energy ratio 8	Time and frequency domain

Figures

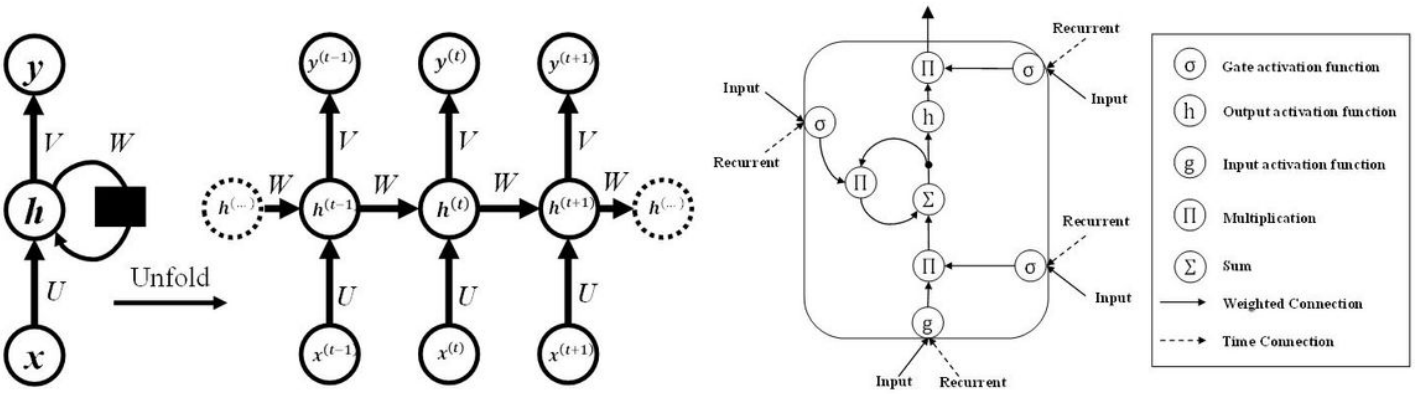


Figure 1

Graphical Depictions of RNN Models. (a) Recurrent Neural Network unfolded in times steps (b) Long Short Term Memory (LSTM) Cell

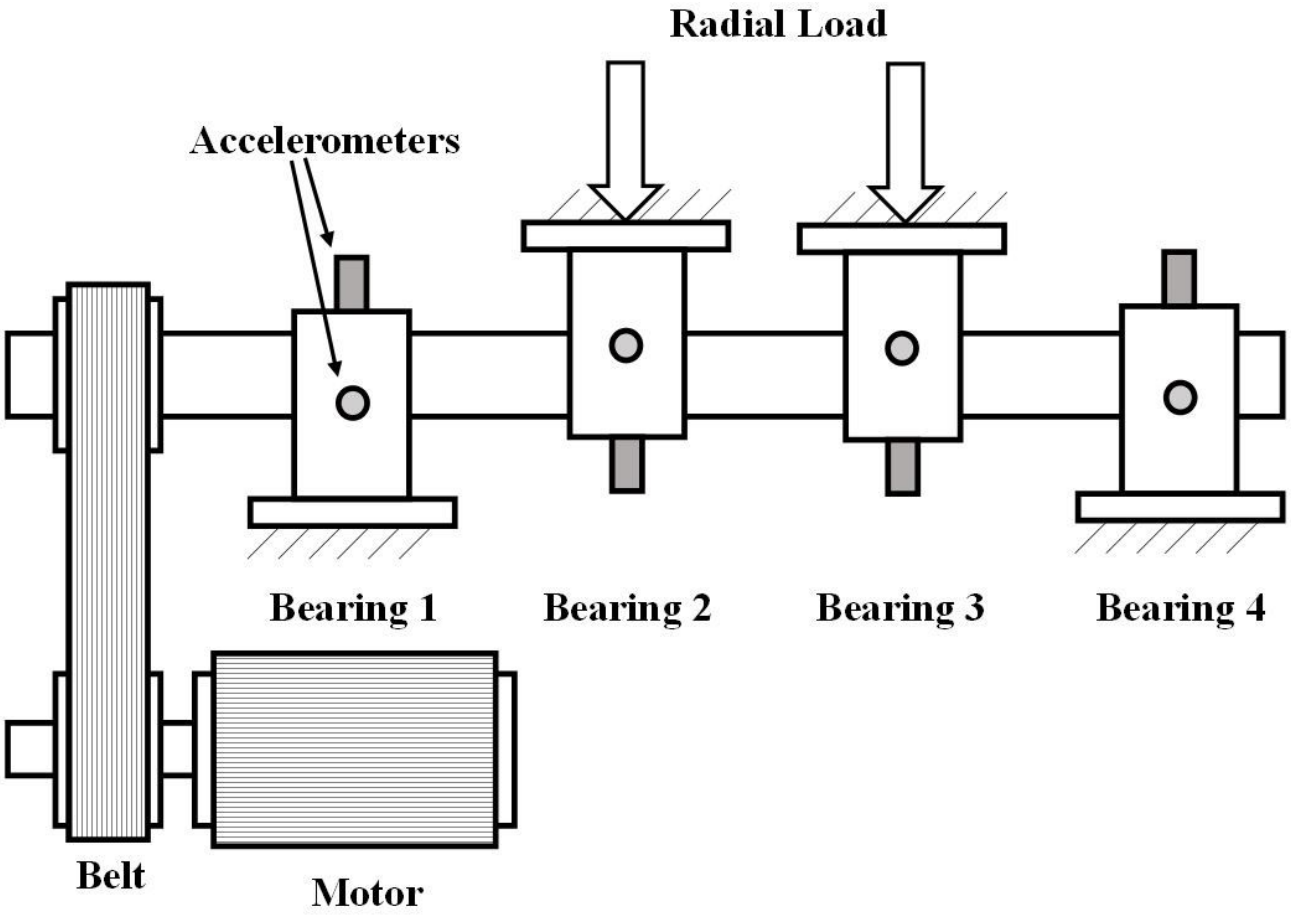


Figure 2

IMS bearing test rig and sensor placement [28]

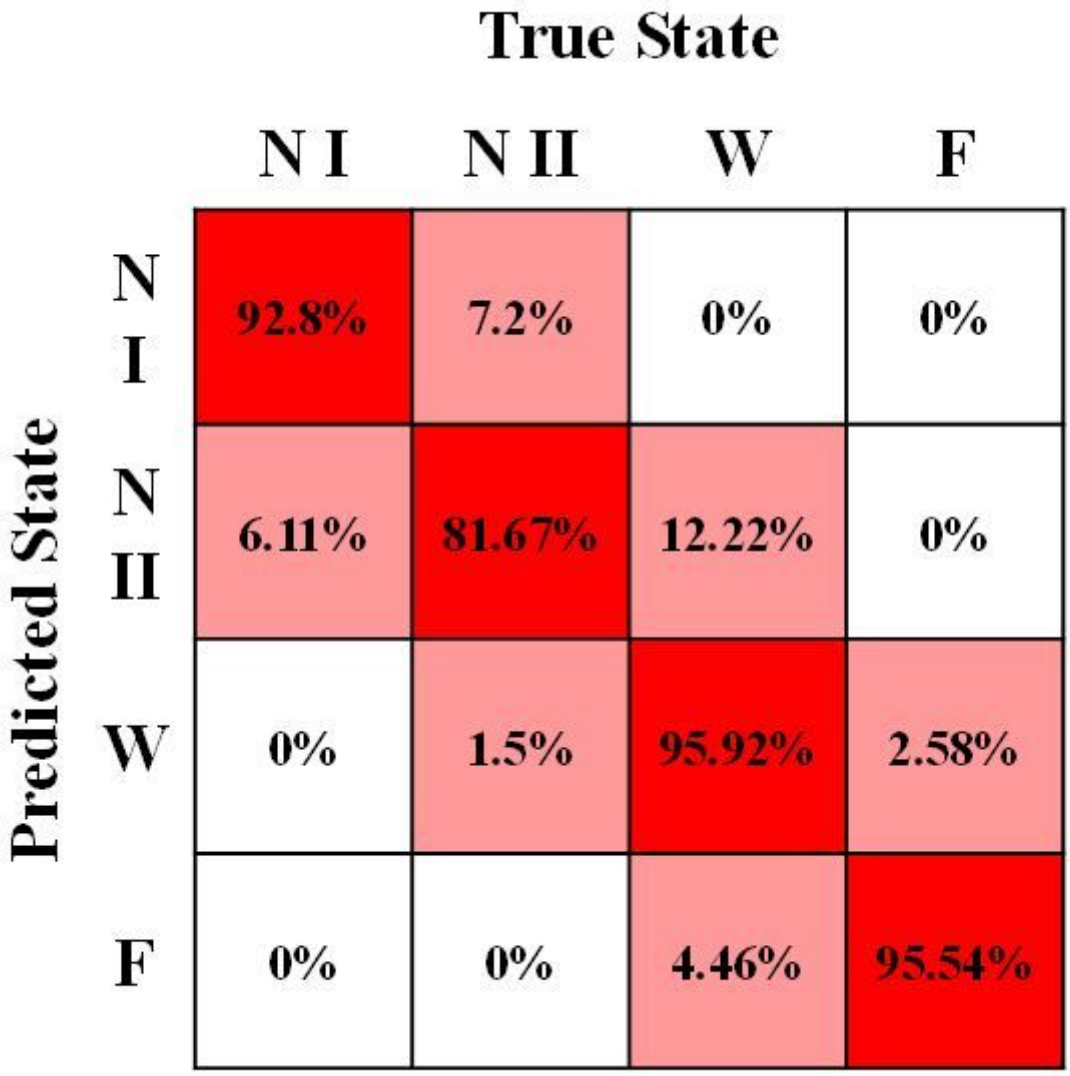


Figure 3

Results from the testing phase of the LSTM-RNN model

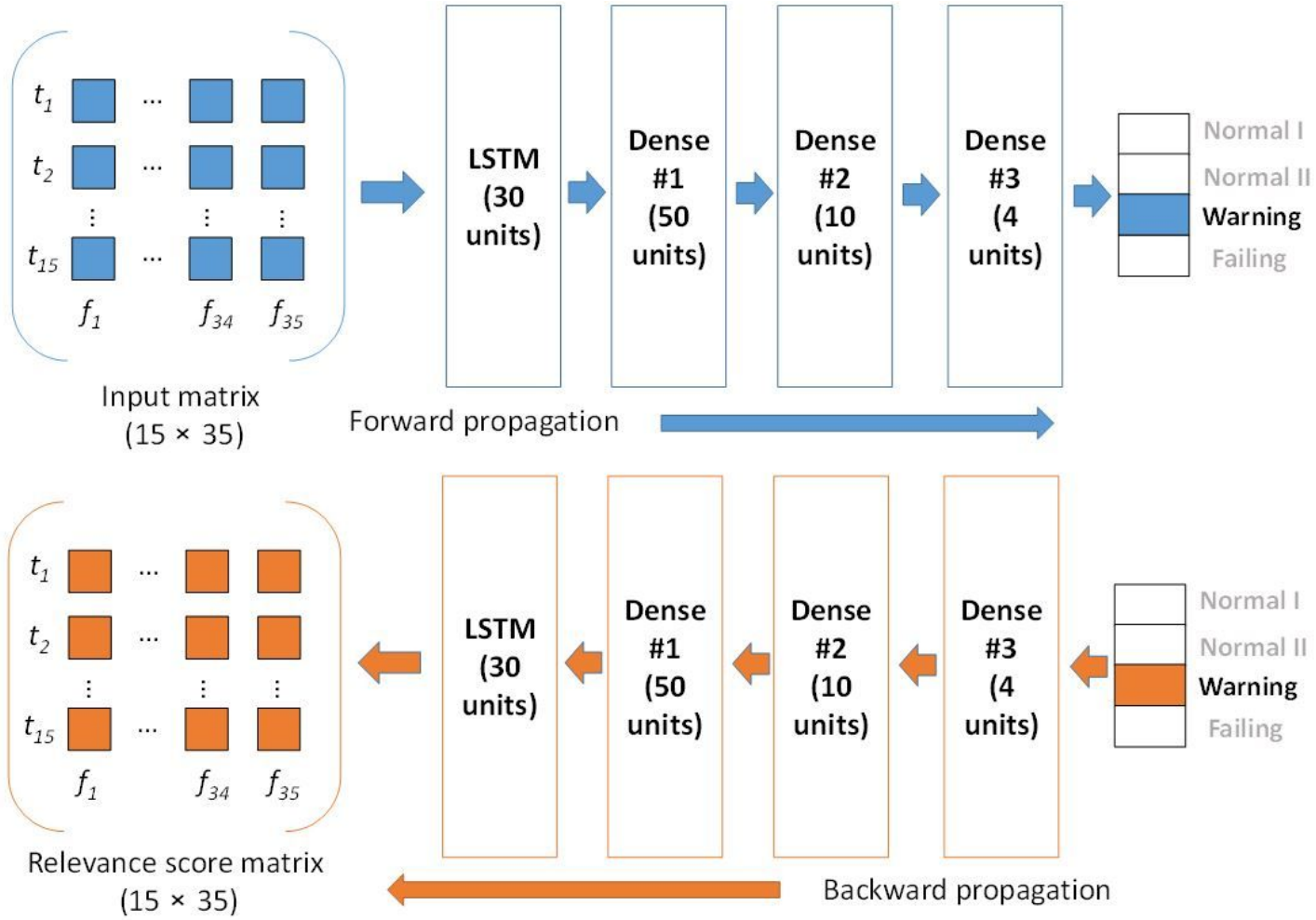


Figure 4

Flowchart for interpreting LSTM-RNN via LRP

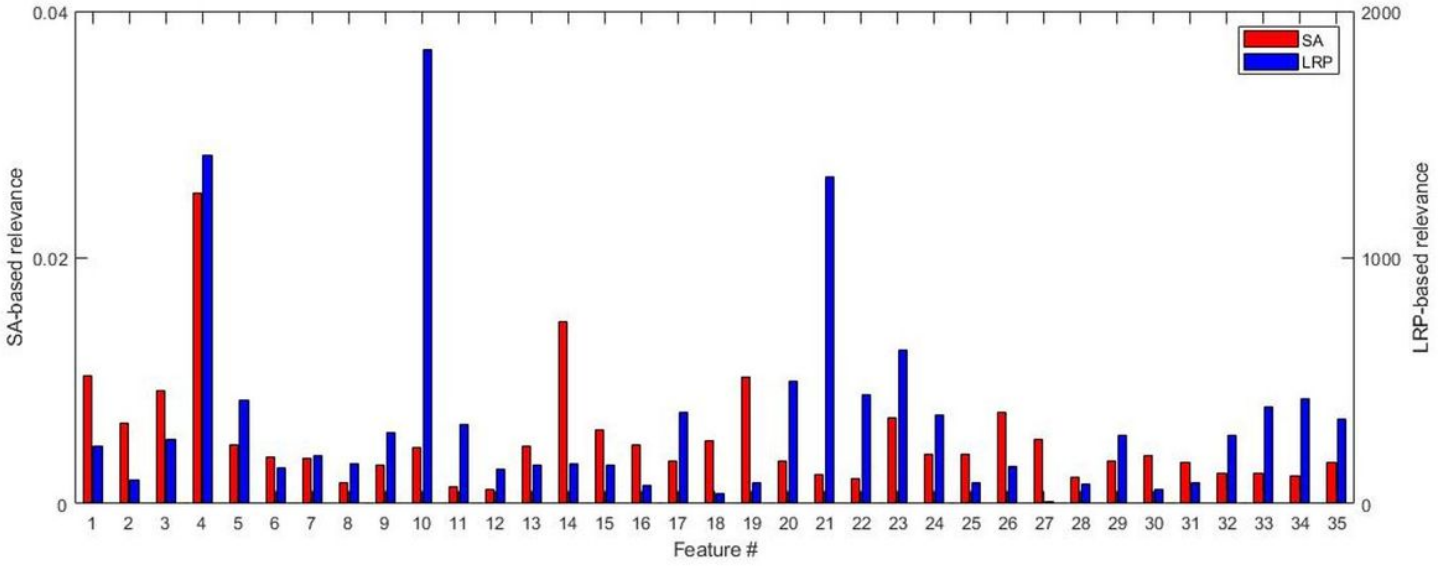


Figure 5

Summed relevance score for features based on LRP and SA

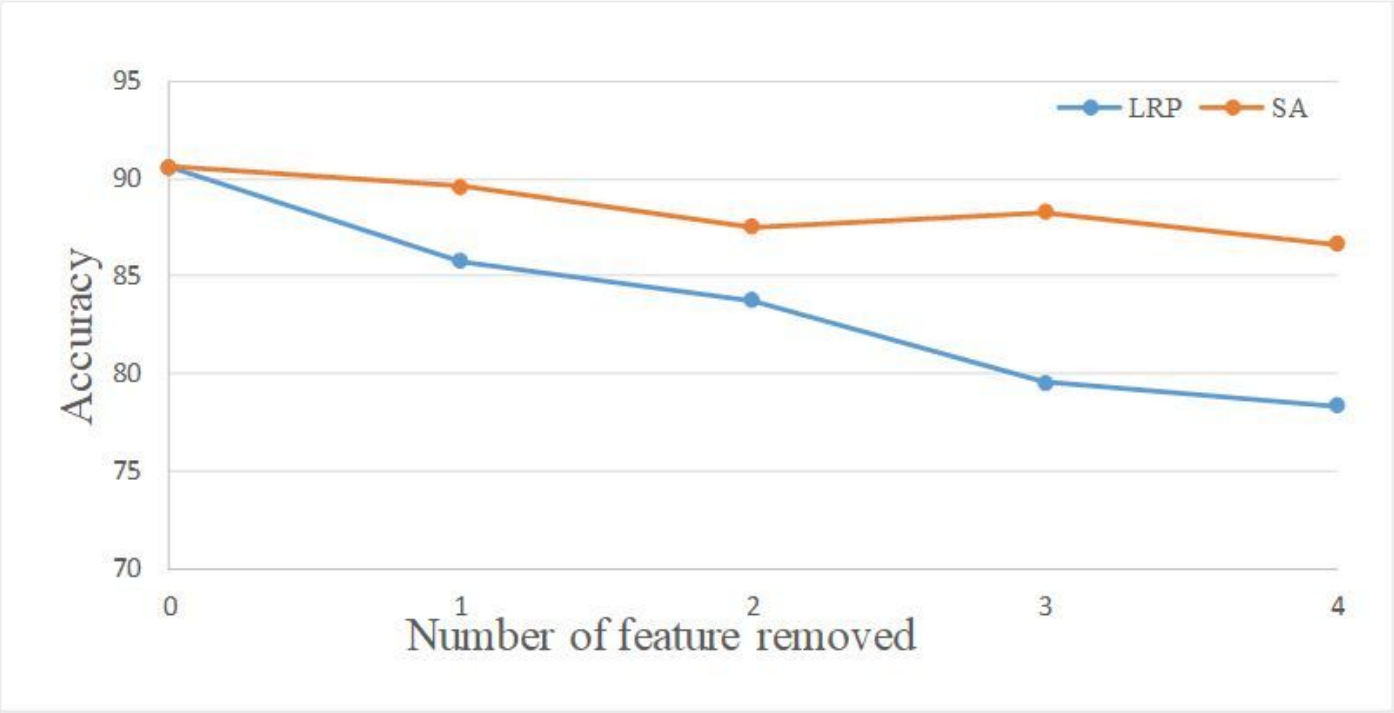


Figure 6

Accuracy loss by removing features according to LRP and SA

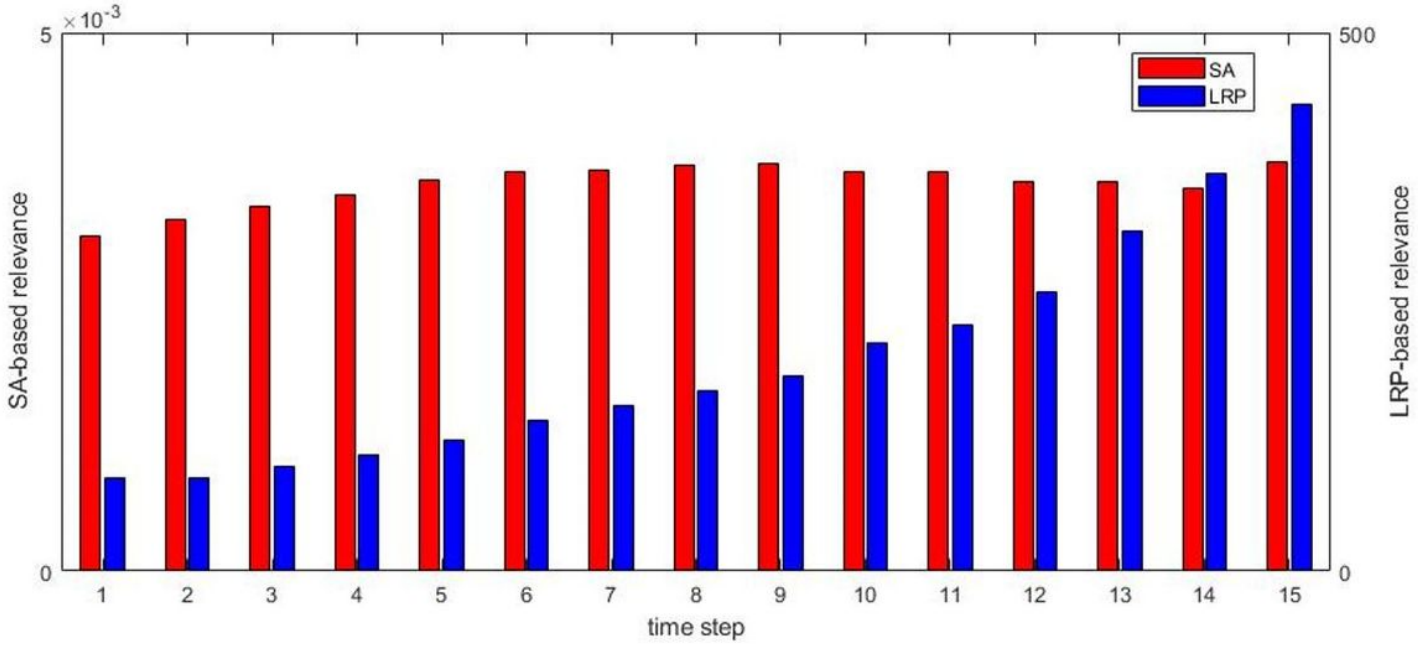


Figure 7

Relevance distribution of time steps

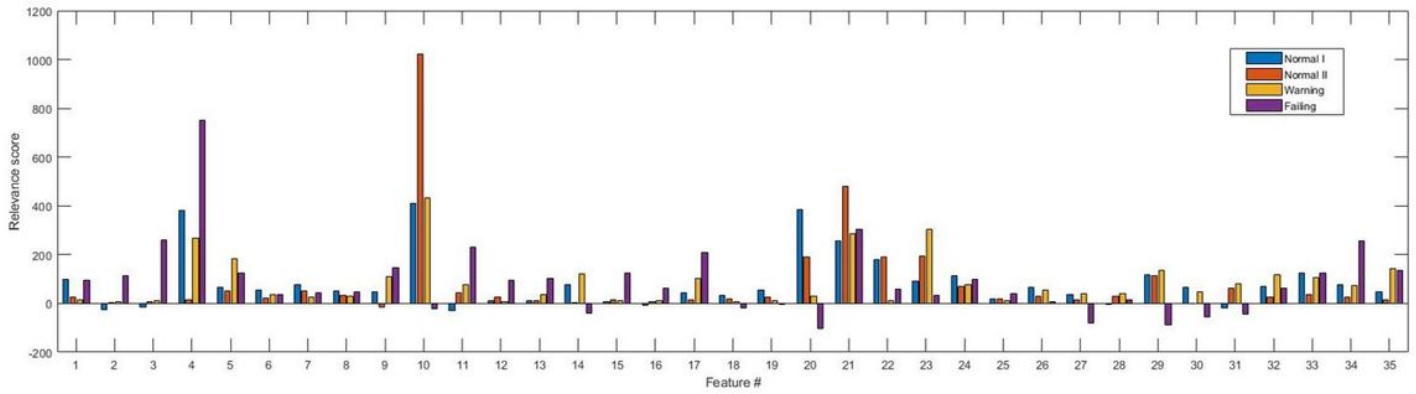


Figure 8

Relevance distribution of features with different classes

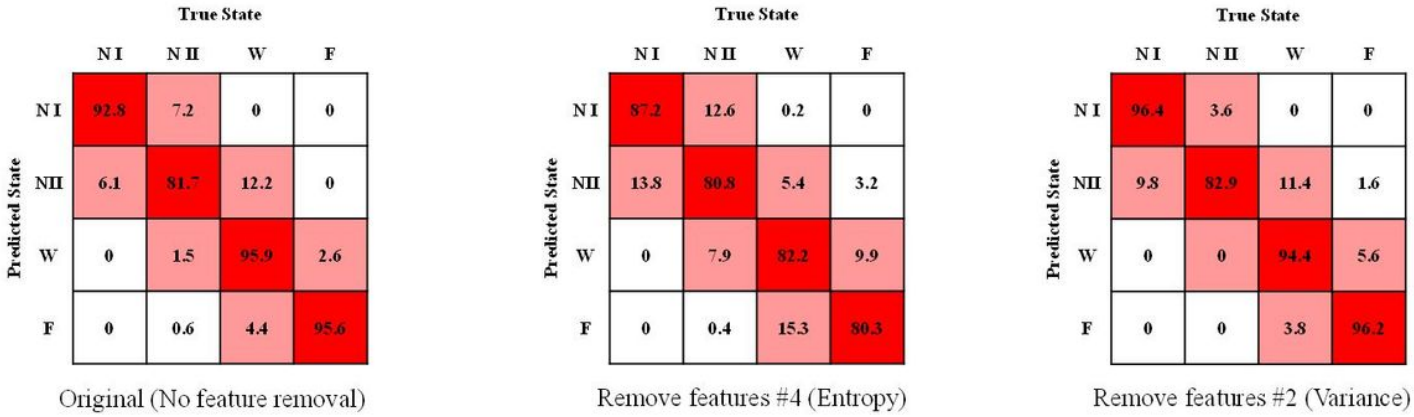


Figure 9

Confusion matrices for the three models

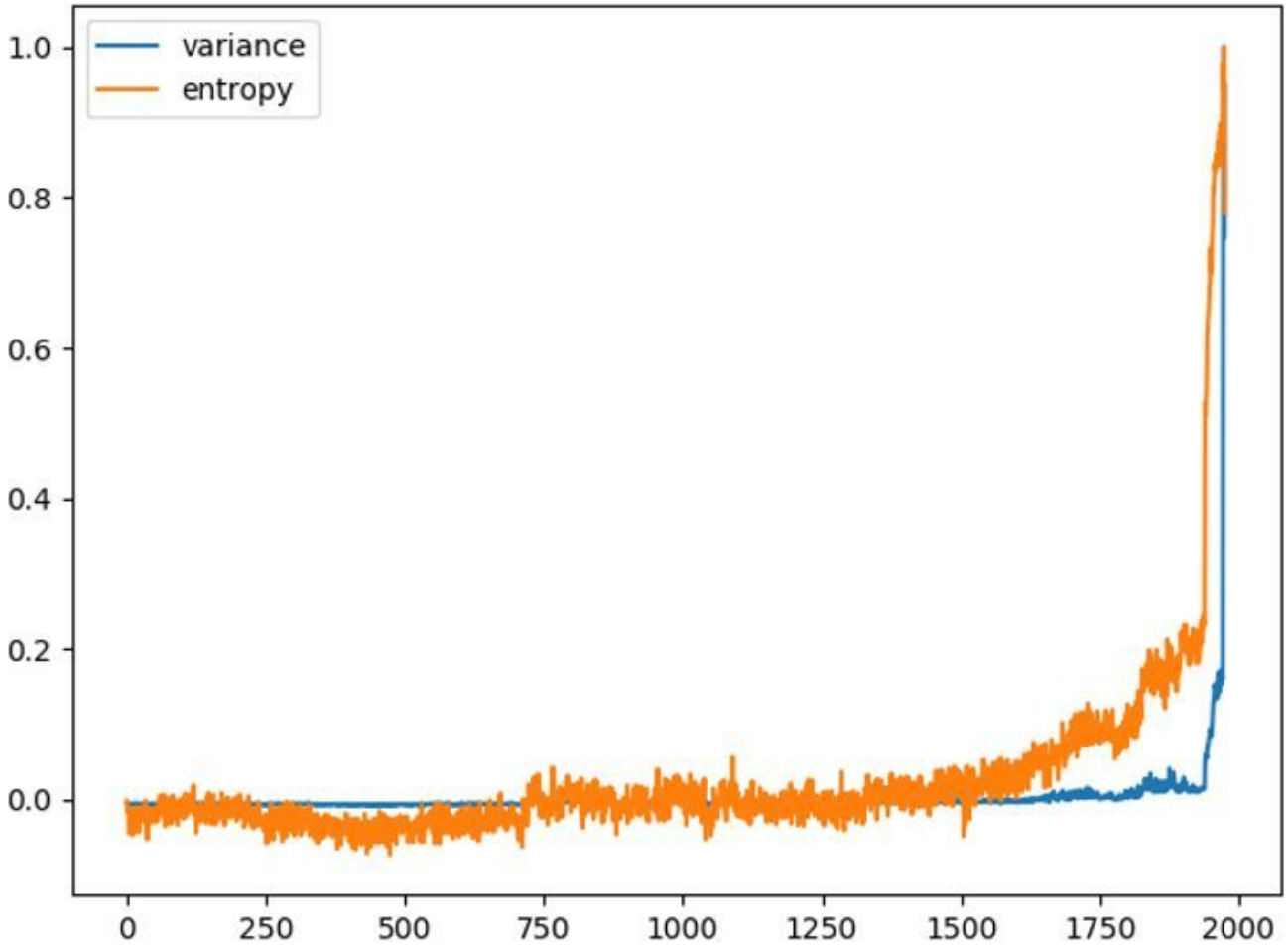


Figure 10

Entropy and variance during the lifespan of a bearing

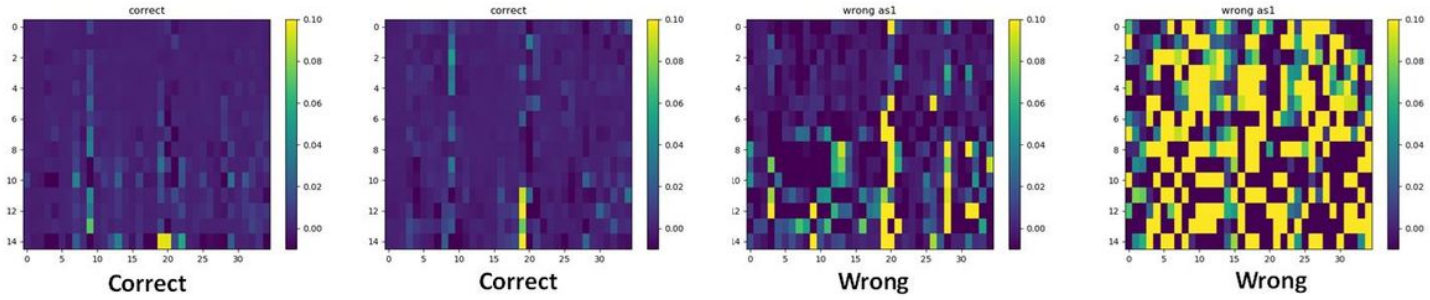


Figure 11

Heat maps of two correctly classified and two wrongly classified samples