Journal of Intelligent Manufacturing (2021) 32:393–406
https://doi.org/10.1007/s10845-020-01578-x

Development of a speed invariant deep learning model with application to condition monitoring of rotating machinery

Wo Jae Lee[1] · Kevin Xia[2] · Nancy L. Denton[3] · Bruno Ribeiro[2] · John W. Sutherland[1]

Received: 5 November 2019 / Accepted: 21 April 2020 / Published online: 9 May 2020

Abstract
The application of cutting-edge technologies such as AI, smart sensors, and IoT in factories is revolutionizing the manufacturing industry. This emerging trend, so called smart manufacturing, is a collection of various technologies that support decision-making in real-time in the presence of changing conditions in manufacturing activities; this may advance manufacturing competitiveness and sustainability. As a factory becomes highly automated, physical asset management comes to be a critical part of an operational life-cycle. Maintenance is one area where the collection of technologies may be applied to enhance operational reliability using a machine condition monitoring system. Data-driven models have been extensively applied to machine condition data to build a fault detection system. Most existing studies on fault detection were developed under a fixed set of operating conditions and tested with data obtained from that set of conditions. Therefore, variability in a model's performance from data obtained from different operating settings is not well reported. There have been limited studies considering changing operational conditions in a data-driven model. For practical applications, a model must identify a targeted fault under variable operational conditions. With this in mind, the goal of this paper is to study invariance of model to changing speed via a deep learning method, which can detect a mechanical imbalance, i.e., targeted fault, under varying speed settings. To study the speed invariance, experimental data obtained from a motor test-bed are processed, and time-series data and time–frequency data are applied to long short-term memory and convolutional neural network, respectively, to evaluate their performance.

Keywords Maintenance · Long short-term memory · Convolutional neural network · Machine condition monitoring · Mechanical imbalance

✉ Wo Jae Lee
lee2465@purdue.edu

Kevin Xia
xia51@purdue.edu

Nancy L. Denton
dentonnl@purdue.edu

Bruno Ribeiro
ribeirob@cs.purdue.edu

John W. Sutherland
jwsuther@purdue.edu

1 Laboratory for Sustainable Manufacturing, Environmental and Ecological Engineering, Purdue University, 500 Central Drive, West Lafayette, IN, USA

2 Department of Computer Science, Purdue University, 300 N. University Street, West Lafayette, IN, USA

3 School of Engineering Technology, Purdue University, 401 N. Grant Street, West Lafayette, IN, USA

Introduction

Starting with the industrial revolution in the eighteenth century, the manufacturing industry has experienced many radical changes such as mass production and system automation. Manufacturing is poised to be changed again with emerging technologies such as artificial intelligence (AI), smart sensors, and Internet of Things (IoT) that seek to establish an integrated and collaborative manufacturing system that responds in real time to changing conditions in the factory. This new trend, i.e., smart manufacturing, is leading the next revolution in the manufacturing industry, and it may enable sustainable growth in the manufacturing sector through the improvement of various manufacturing performance measures such as energy efficiency, quality, and productivity (Kim 2019). As one example of a national effort to capitalize on new technologies, the U.S. Department of Energy's Advanced Manufacturing Office (AMO) launched the Clean Energy Smart Manufacturing Innovation Institute (CESMII) to advance the country's manufacturing competitiveness and reduce its environmental impact (CESMII). The institute supports research and development of technologies that can collect, share, and process the huge amount of data obtainable from manufacturing activities in real-time.

One application of smart manufacturing is an intelligent maintenance system (Lee et al. 2019a). A goal of the maintenance is maximizing the availability of manufacturing systems to increase productivity while reducing maintenance cost by (1) optimizing maintenance tasks and (2) fixing potential defects before catastrophic equipment failures occur, i.e., prevent unplanned downtime. To enable this, the condition of equipment needs to be continuously monitored without interruption (non-intrusive monitoring), and future behavior must be predicted (e.g., prognostic health management) (Seevers et al. 2019). With the present proliferation of sensing and communication technologies available in a production line, extensive machine condition data may be collected in many factories. The condition data are normally proxy measures (e.g., vibration, acoustic emission, and temperature). Thus, a method is required to extract meaningful information, e.g., health condition, from large-scale condition data available from operating equipment.

Condition monitoring methods are often classified into three categories: (1) a physical model, (2) a knowledge-based model, and (3) a data-driven model (Peng et al. 2010). A physical model-based methodology normally shows good success at reflecting the condition of the monitored system because the model is built based on accurate mathematical relations tied to physical processes. However, establishing an accurate physical model is challenging for complex manufacturing systems. Also, a physical model cannot generally be updated with on-line measurement data, which limits the model's flexibility (Zhao et al. 2019). A knowledge-based methodology, such as an expert system, solves a specific domain problem using expert knowledge and heuristic rules. In this methodology, an accurate physical model is not required, but translating domain knowledge into rules (e.g., IF conditions) is difficult and the model may not cope well with new situations. Lastly, a data-driven model estimates model parameters to fit the model using input and output data. This method is based on statistical learning theory, and the model automatically learns a relationship between input and output data (supervised learning) during the training phase. However, the method often requires a large amount of machine condition data for model training and testing.

Among the methods, data-driven models [e.g., artificial neural networks (Jia et al. 2018) and random forest (Pimenov et al. 2018), kernel principal component analysis (Lee et al. 2019b)] have received a great deal of attention by researchers due to increasing availability of open source data and advances in computing infrastructure (e.g., GPUs). Recently, deep learning (DL) methods, which originated from artificial neural networks (ANN), have been applied extensively to machine condition datasets for health condition monitoring research. Janssens et al. (2016) proposed an automatic bearing fault detection method using convolutional neural networks (CNN). In the study, different types of bearing faults (e.g., outer-raceway fault and rotor imbalance) were detected using acceleration signals obtained for a 25 Hz rotational speed. Jing et al. (2017) also used a CNN for condition monitoring of gearboxes. They compared model prediction accuracies using both automatically learned features and manually extracted features. A number of CNN network configurations (e.g., various filter sizes, numbers of filters, and numbers of convolutional layers) were tested. Cacciola et al. (2016) studied a neural network-based monitoring system to identify different root causes of mechanical imbalance problems in a rotor. Jia et al. (2016) showed an improved performance of deep neural networks compared to shallow neural networks for the diagnosis of the bearing and planetary gearboxes using an auto-encoder for data preprocessing. The DL-based monitoring approach was reported to be superior to classical machine learning techniques (e.g., Support Vector Machine (SVM) and random forest) (Jing et al. 2017). Khan and Yairi (2018) summarized various DL methods and their applications to system health monitoring. They concluded that there is a growing interest in applying DL methods in the engineering community, but many limitations still exist such as design, selection, and implementation of DL methods.

As is evident from the literature review, DL is an evolving and growing area for machine condition monitoring research, and its ability to predict conditions offers substantial promise (some people may argue that DL applications to mechanical diagnosis and prognosis are still lacking when compared to other fields such as speech recognition and image classification). One may think that, because training a DL model is computationally expensive, it may not be suitable for manufacturing applications. However, recently, there have been significant advancements in the DL research field to overcome shortcomings by reducing connectivity in networks (e.g., CNN) and developing an efficient training method (e.g., Adam optimizer). One attractive advantage of DL is reducing the amount of effort for feature engineering by learning non-linear representations in a large dataset using multiple hierarchical layered structures. This may enable the model to predict a targeted fault, in which an indicator relating to a target fault is non-linearly correlated to a machine health condition. Such a model may possess the ability to detect and locate a fault in sophisticated manufacturing equipment.

Several types of popular network architectures (e.g., CNN and recurrent neural network) and their variant were widely applied on the machine condition data, and their performances on the machine fault diagnosis were evaluated and reported in the cited paper. However, work related to a model's response to data obtained from operating conditions that differ from the training data has not been extensively examined, i.e., a trained model may work well only for data obtained from a certain operating setting. Therefore, there is a lack of studies focusing on a performance variation of a deep learning model, which has already been tuned with the data obtained from a certain operational condition, to the data collected from the different operational conditions. Although DL is known to be a powerful tool to automatically learn and discover representations needed for classification from large-scale datasets (called representation learning), it may be difficult for a DL model to detect a targeted fault when analyzing data collected under previously unseen operating settings. Because machine operating settings can change during the manufacturing process, a model's performance should be invariant to a variable operating environment (e.g., variable rotational speed) while monitoring a system. Park et al. (2019) argued that previous works on condition monitoring mainly focused on detecting a fault under constant rotational speed although many real-world applications run under variable speed. Accordingly, DL applications need to be further studied and tested with data obtained from different operating conditions (e.g., different rotational speeds (RPM)) as well as using various types of machine condition data (e.g., acceleration and acoustic emission). Ultimately, a method that is invariant to changes in rotational speed (RPM) must be considered for practical applications. The main contributions of this paper are: (1) the idea/property of invariance to speed change is discussed, (2) a DL based mechanical imbalance monitoring system is proposed, (3) an improved long short-term memory (LSTM) model is developed using an attention mechanism, (4) performance variations of deep learning models (CNN and LSTM) to the data obtained from the different operational conditions are examined and compared using experimental data, and (5) the effectiveness of the proposed method (Scaled and Smoothed TS-LSTM with Attention) is demonstrated.

The paper is organized as follow. First, the property of RPM invariance is mathematically explained, which defines detection accuracy invariance to varying RPM. Then, data preprocessing methods, which will be combined with deep learning models, are proposed. For DL architectures, LSTM and CNN are employed to detect a targeted fault, and their basic theories and the customized architectures are explained. To experimentally study the RPM invariance in a deep learning model, sets of experiments were conducted using a motor testbed. During the experiment, machine condition data were collected using a triaxial accelerometer under various RPMs at certain mechanical imbalance levels. Then, raw signals are processed to extract features and the features are applied to evaluate DL models' performances under both constant RPM and varying RPM conditions. Performance variations in DL models are reported using the data obtained from previously unseen RPM settings during the training phase (i.e., test a model with the data obtained at different rotational speeds). All data collected from the experiments reported on herein will be available via the Purdue Laboratory for Sustainable Manufacturing (LSM).

Invariance to changing rotational speed in fault detection

The goal is to establish a fault detection model whose accuracy is invariant to changes in the RPM (we will refer to this as 'RPM invariance'). This will be accomplished by predicting a targeted fault condition using a proxy measure, e.g., vibration, in a motor system that runs at previously unseen RPMs. Given motor vibration data points (either raw or processed) $\boldsymbol{x} \in \Omega$, the function of interest is $f : \Omega \rightarrow \{1, 2, \ldots, N\}$ which maps the data points to the corresponding fault condition $y = f(\boldsymbol{x})$ when N conditions are defined. The shape of the sample space, Ω, varies depending on the format of the data. The function, f, is approximated using a data driven model (e.g., neural network model), $\hat{f}_\theta$, parameterized by θ to make a prediction of y, $\hat{y} = \hat{f}_\theta(\boldsymbol{x})$. The rotational speed of a motor can be described as a function of the data collected from that motor, defined as $r : \Omega \rightarrow \mathbb{R}^+$ such that $r(\boldsymbol{x})$ is the RPM of data points $\boldsymbol{x} \in \Omega$. Then, the notion of RPM invariance can be defined as follows. Given $\boldsymbol{x} \in \Omega$ and $\alpha \in \mathbb{R}^+$, let $s : \Omega \times \mathbb{R}^+ \rightarrow \Omega$ such that $s(x, \alpha) = x'$, where $f(\boldsymbol{x'}) = f(\boldsymbol{x})$, $r(\boldsymbol{x'}) = \alpha r(\boldsymbol{x})$, and α is the ratio of desired (testing) RPM to current (training) RPM. Here, $\boldsymbol{x'}$ is an RPM transformation of $\boldsymbol{x}$ by α. Then, the property of RPM invariance for $\hat{f}_\theta$ is

$$\hat{f}_\theta(s(\boldsymbol{x}, \alpha)) = \hat{f}_\theta(\boldsymbol{x}) \tag{1}$$

for all $\boldsymbol{x}$ and α. This property means that changing the RPM of the data should not affect the prediction of the model. To achieve this, this paper focuses on the details of r and f, and finding a procedure for determining them.

Data preprocessing and deep learning models

Once the data acquisition plan (e.g., sensor type, sampling rate, and data acquisition interval) is decided, a sensor can be mounted on manufacturing equipment, and raw sensor signals may be collected for a certain machine health condition. Then, the collected signals, i.e., machine condition data, can be processed to generate features, which may better represent the machine health condition. In case of vibration signals, features from the time, frequency, and time–frequency domain data are often used for deep learning (DL) applications (Zhang et al. 2013). Also, in order to analyze a non-stationary vibration signal, order analysis or order-tracking method were often used to extract vibration data related to the rotational speeds. However, during the experiment in this study, a range of rotational speed was not very wide, and a speed was not increased continuously (i.e., increased from 300 to 380 in 20 RPM increments). Thus, features from order analysis may not be useful. Instead, other data preprocessing methods, which will be described in this section, are employed in this paper.

In this section, two data preprocessing methods and two DL architectures are introduced to study the RPM invariance in a DL model. In "Scaling and smoothing of time-series data obtained from different RPM settings" section, a method which may have the properties of RPM invariance in a LSTM model is proposed first. A second data preprocessing method, i.e., continuous wavelet transform (CWT), which will be combined with CNN, is explained in "Extracting time–frequency features using continuous wavelet transform" section. CWT is a technique to extract time–frequency features from vibration signals, and CWT has been often combined with CNN models (Yoo and Baek 2018). Therefore, this method may be a good candidate to compare with the first approach.

In "Deep learning models using LSTM and CNN" section, the basic theory and proposed architecture of two deep learning methods, LSTM and CNN, are explained. The selection of a model is dependent on the type of data being analyzed. For data collected over time (e.g., time series data), a recurrent neural network architecture is often used, specifically a long short-term memory (LSTM) model (Zhang et al. 2018). For data arranged in a matrix such as time–frequency data (e.g., short time Fourier transform and wavelet transform), a convolutional neural network (CNN) model is normally used (Verstraete et al. 2017). Therefore, in this paper, time-series data and time–frequency data are used to evaluate the LSTM-based model and the CNN-based model, respectively.

Data preprocessing for vibration signal

Scaling and smoothing of time-series data obtained from different RPM settings

A change in the speed of equipment with rotational elements almost always leads to changes in the frequency content of vibration sensor signals. For example, increasing the RPM may shift the dominant frequencies in the frequency domain to larger values. By the time scaling property of the Fourier transform, a scale in the frequency domain corresponds to an inverse scale in the time domain. Hence, if the test data RPM is different from the RPMs for the training data, then one may expect that a transformation of the test data would better match the vibration frequencies observed in the training data.

However, in practice, not all vibrations captured in the sensors are related to the RPM of the motor. Other factors like fluid flow, electrical components, and non-rotating elements all affect vibrations. Furthermore, high frequency noise tends to obscure the structural content in the data. To remedy this, a noise-reducing data transformation is implemented for raw data that may mimic data collected from other RPM settings. This procedure, visualized in Fig. 1, involves: (1) scaling the time-domain data by the ratio α using a spline interpolation, (2) converting the data to the frequency domain using the discrete Fourier transform (DFT), (3) filtering out high frequency components using a low pass filter and removing less significant amplitudes, and (4) converting back to the time domain using the inverse DFT. The frequency removal acts as a smoothing procedure, removing some of the abrupt changes in the data, and is applied to all data used in the models regardless of whether the RPM needs to be changed.

Extracting time–frequency features using continuous wavelet transform

Time–frequency analysis transforms a signal in the time domain, $x(t)$, to the time–frequency domain, in which various frequency components are present over time (e.g., short time Fourier transform). Unlike the short time Fourier transform which generates time–frequency representations in the fixed frequency resolution, a wavelet transform creates a frequency-dependent frequency resolution using a scalable window function called the mother wavelet (ψ) (Verstraete et al. 2017). Given a wavelet function, $\psi(t) \in L^2(\mathbf{R})$, which has nonzero values only in certain range, the continuous wavelet transform is written as

$$\mathrm{W}_x(\tau, \mathrm{s}; \psi) = \frac{1}{\sqrt{|s|}} \int_{-\infty}^{\infty} x(t)\psi^*\left(\frac{t-\tau}{s}\right) dt \tag{2}$$

where τ, s, and $\psi^*(\cdot)$ are the translation parameter, scale parameter, and the complex conjugate of $\psi(\cdot)$, respectively. Here, the signal is convolved with a scaled wavelet, thus $\mathrm{W}_x(\tau, \mathrm{s})$ represents the degree of correlation between the signal and the wavelet given τ and s (Park et al. 2019). Because the wavelet transform enables multi-scale analysis of a signal using the two variables, τ and s, it can effectively extract time–frequency features from nonstationary and transient signals (Peng and Chu 2004). In this paper, the Morlet wavelet is employed, which is mathematically expressed as

$$\psi(t) = e^{-t^2/2}\cos(5t). \tag{3}$$

Wavelet transforms have been extensively used to extract time–frequency features and combined with various machine learning techniques in condition monitoring research (Peng

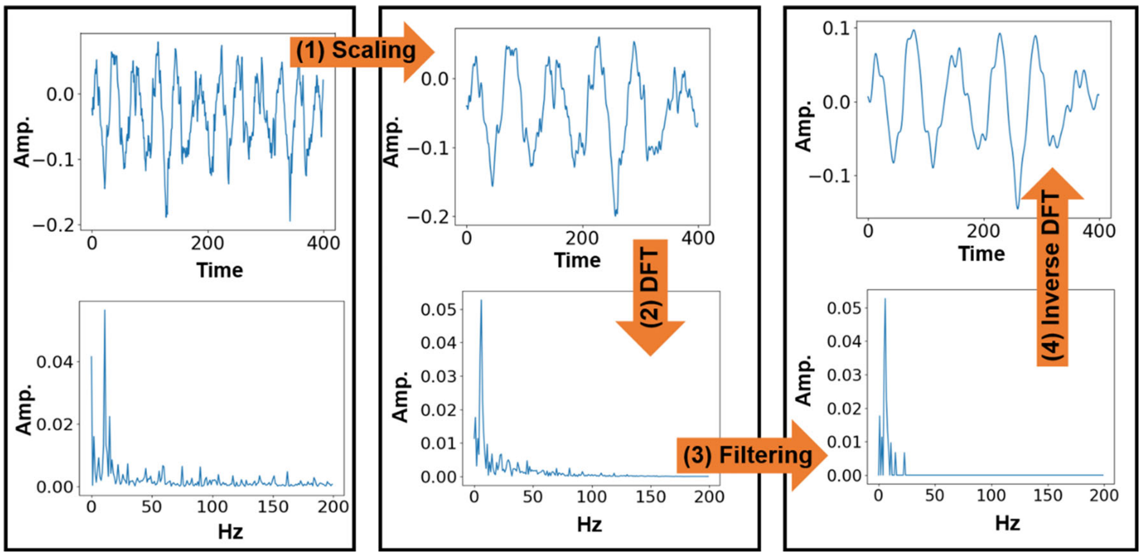


Fig. 1 Raw data transformation procedure visualized in both the time and frequency domains

and Chu 2004; Zhang et al. 2013). However, the previous studies mainly focus on detecting a targeted fault for a constant speed condition. In the present work, however, as explained before, DL models will be trained with features extracted from CWT, and then evaluated using experimental data obtained from varying RPM settings and compared with the method explained in "Scaling and smoothing of time-series data obtained from different RPM settings".

Deep learning models using LSTM and CNN

Long short-term memory (LSTM)

An LSTM follows a recurrent architecture, that is, outputs from one layer can serve as inputs for the same layer, allowing information to persist across entire sequences of inputs. A typical LSTM architecture is shown in Fig. 2 (Colah).

The LSTM can be distinguished from other recurrent neural networks by its use of gates. Specifically, with each pass through the recurrent layer of the LSTM, depicted as "G" in Fig. 2, the following may be computed:

$$d_t = \sigma(W_f \times (h_{t-1} \oplus x_t) + b_f), \tag{4}$$

$$i_t = \sigma(W_i \times (h_{t-1} \oplus x_t) + b_i), \tag{5}$$

$$c_t = d_t \cdot c_{t-1} + i_t \cdot \tanh(W_c \times (h_{t-1} \oplus x_t) + b_c), \tag{6}$$

$$o_t = \sigma(W_o \times (h_{t-1} \oplus x_t) + b_o), \tag{7}$$

$$h_t = o_t \cdot \tanh(c_t), \tag{8}$$

$$\sigma(x) = \frac{1}{1 + e^{-x}}. \tag{9}$$

Here, the operation symbols, $\times$, $\oplus$, and $\cdot$, represent the matrix multiplication, the concatenation, and the element-wise multiplication, respectively. σ and tanh represent the sigmoid function and the hyperbolic tangent, which has output values between 0 and 1 and between -1 and 1, respectively. W and b are learnable weights and biases.

A qualitative explanation can be provided for each of these gates. d is the forget gate, with values between 0 and 1 that determines how much of the previous state to retain. i is the input gate, with values between 0 and 1 that determines how much of the input to accept. c is the cell state, which can be described as the memory of the layer. It uses the forget and input gates to determine how much information to retain and change between iterations. o is the output gate, which has values between 0 and 1 that determines how much of the cell state to pass to the output. Finally, h is the hidden state, which is passed as the output to the next layer and is also passed back into the same layer for the next iteration. It is simply the cell state filtered by the output gate.

The architecture used in this paper combines one of these LSTM layers with the attention mechanism (Vaswani et al. 2017) and a fully connected layer. After passing the data through the LSTM layer, the vector of hidden states, h, is passed through an attention mechanism described by the following equations.

$$q = \tanh(W_h \times h + b_h) \tag{10}$$

$$\beta = \mathrm{softmax}(W_q \times q + b_q) \tag{11}$$

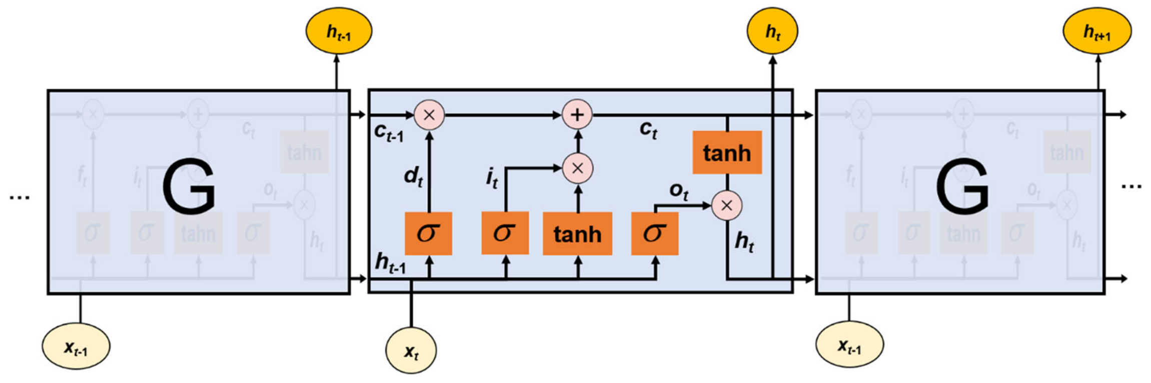


Fig. 2 A typical LSTM architecture

$$h_{att} = \sum_{i=1}^{m} \beta_i h_i \tag{12}$$

Once again W and b are learnable weights and biases. The softmax activation function is a function that normalizes a vector so that all its values are all positive and sum to 1. Qualitatively speaking, q represents a learned embedding for each of the hidden states, β is a normalized weight vector that assigns an importance value to each of the hidden states, and h_{att} is an average of the m hidden states weighted by β. Finally, h_{att} is passed through a fully connected layer for the final prediction.

The attention mechanism in the LSTM model can be used to overcome limitations of long sequential data by determining how much "attention" should be paid to each time step in the hidden state. A typical LSTM model uses only the hidden state information from the final time step, often causing information from earlier iterations to be forgotten. However, attention uses information from all time steps of the hidden state, prioritizing the ones that are most important for classification, so important information in the past is not lost.

For time series data, the input is three stacked time series—vibration data from the X, Y, and Z directions, i.e., processed data from a triaxial accelerometer, each with 400 time-steps as shown in Fig. 3 (this architecture is called TS-LSTM in this paper). The model (1) passes each of the 400 time-steps through the LSTM layer, (2) takes the hidden states (from h_1 to h_{400}) from the entire pass, (3) multiply with the outputs from the attention mechanism, and (4) feeds them through a 128-length fully-connected layer for classification. The fully-connected layer outputs a value between 0 and 1, which is rounded to produce the predicted targeted fault. The number in gray rectangular (e.g., 128×128) means there are 128×128 connection between layers.

Convolutional neural network (CNN)

A CNN consists of alternating convolutional layers and pooling layers, followed by a fully connected layer. Convolutional layers use several filters, each mapping the input matrix to an output matrix. Filters take small regions of the input, multiply them by learned weights, and pass the result to the output. Formally, a convolutional layer convolves input X_{in} as described in Eq. (14), where W_k and b_k are the weights and bias of the k th filter, and g is the activation function, often the rectified linear unit (ReLU).

$$g(x) = \max(0, x), \tag{13}$$

$$X_{out,k} = g(X_{in} * W_k + b_k). \tag{14}$$

In the equation, * represents the convolution operator, where an output matrix is produced by applying the kth filter across all regions of the input. All outputs from all filters are stacked to produce the input for the next layer. This final output is often called a feature map since the values represent features of the original input.

Pooling layers reduce the dimensionality and spatial precision of the input by sub-sampling the input. Pooling locally combines each window of the input into a single value in the output. In this paper, max pooling (i.e., max filter) is used, so each value in the output of the pooling layer corresponds to the maximum value of a small region in the input.

Finally, after multiple convolutional and pooling layers, the resulting output is passed through a fully connected layer. Mathematically, this can be written as:

$$X_{out} = g(W_j X_{in} + b_j). \tag{15}$$

Equation (15) describes the effect of the fully connected layer on the input X_{in}, where W_j and b_j are the weights and bias respectively for the j th output. Here, g is an activation function, and ReLU is used as the activation function for all

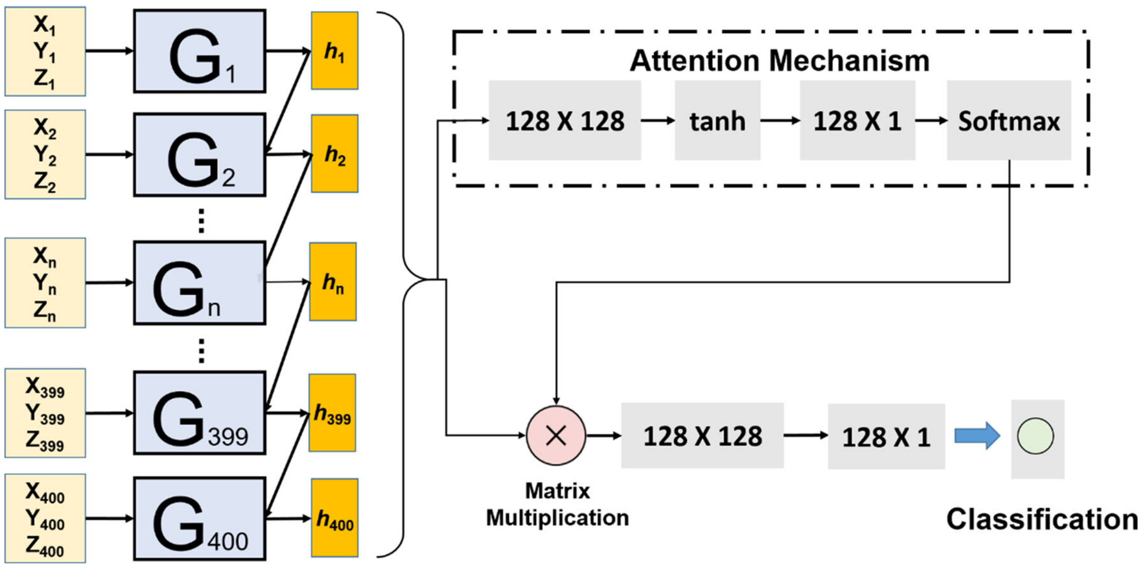


Fig. 3 A proposed LSTM architecture for time-series data (TS-LSTM)

layers except for final layer, for which the sigmoid function is used.

In this paper, time–frequency data are applied to a CNN model; the proposed CNN architecture is shown in Fig. 4 (called CWT-CNN). The figure is generated using NN-SVG (NN- SVG). The input is three matrices, representing the amplitudes in the X, Y, and Z directions across 400 time-steps and 311 frequencies. The output is a value between 0 and 1, which is rounded to produce the predicted targeted fault.

Mechanical imbalance experiment and condition data acquisition

A mechanical imbalance can be defined as an uneven distribution of mass/force about a rotating centerline. In a motor system, where power is transmitted from a motor to driven equipment, some level of mechanical imbalance is always present owing numerous factors, e.g., rotor wear/damage, debris buildup, manufacturing and assembly variation, and poor design (Cacciola et al. 2016). When this imbalance becomes large, it may affect the performance of the motor system. A mechanically unbalanced motor system may experience rapid wear on mechanical components (e.g., bearings), and consequently lead to a shorter life span of manufacturing equipment. Failure of mechanical components can often be traced to system imbalance (and, in turn, the imbalance is often attributable to other causes), so it is generally prudent to detect an imbalance and take corrective actions as early as possible.

To collect acceleration signals for different imbalance conditions in a motor system, experiments were conducted using a motor testbed. Overall configuration of the testbed is shown in Fig. 5a. The testbed is equipped with a ¼ horsepower motor with pulse width modulation variable speed DC drive. To induce different levels of mechanical imbalance in the testbed, two planar balancing disks with 24 equally spaced holes are mounted on a shaft between two bearing supports. To create an imbalance condition during some of the tests, two masses were mounted to the disk as shown in Fig. 5b (the masses of ① and ② in Fig. 5b are 27.06 g and 29.08 g, respectively). The photo tachometer (Extech, 461895) was used to measure motor speed during the experiment (Fig. 5c). A triaxial accelerometer (PCB PIEZOTRONICS, J356A45) was attached using adhesive as shown in Fig. 5d.

In the experiment, two levels of the mechanical imbalance were introduced. The two levels are a "balance" or default condition (no masses on the disks) and imbalanced conditions (with the two masses added to the disks). During the experiment, the motor speed (RPM) was increased from 300 to 380 in 20 RPM increments, and triaxial acceleration signals were collected using a National Instruments (NI) Compact Data Acquisition System that included chassis (NI, cDAQ-9178) and Sound and Vibration Input Module (NI, NI-9234). The data were collected under steady-state operating conditions for a total of ten data sets (two levels of imbalance and five rotational speeds, 300, 320, 340, 360, and 380 RPM). LabVIEW software was used to store the sensor signals in a PC, and the sampling rate for the X, Y, and Z channels was set to 3.2 kHz. The digital data were sampled 50 times at 10-s intervals (i.e., $3200 \times 3 \times 50 = 480{,}000$ data for one set).

Figure 6 shows examples of the typical triaxial acceleration signals obtained from different imbalance and operating conditions and their corresponding spectra using FFT. Also, in order to numerically compare the differences in the tri-

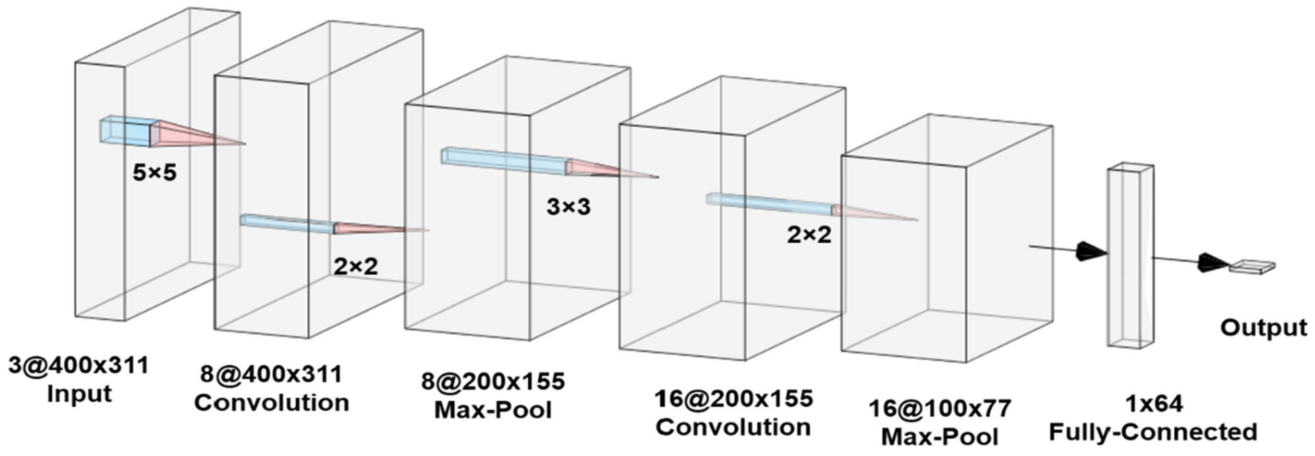


Fig. 4 A proposed CNN architecture for time–frequency data (CWT-CNN)

axial acceleration signals, two features, root mean square (RMS) and Kurtosis, which were extracted from the time domain and frequency domain, are computed in Table 1. As expected, the longitudinal direction (Y axis) displays a smaller vibration than other directions in terms of RMS value during the operation because there was no significant movement in the longitudinal direction in the testbed. While the highest RMS values were observed in X axis, Z axis shows the greatest Kurtosis values in most cases which means that a heavier tail exists over the frequency distribution. As seen in the table, a distinguishable pattern can be found in each axis as rotational speed and load-setting change.

Despite noticeable differences among the acceleration signals shown in Fig. 6, it may be hard to manually distinguish a mechanical imbalance condition by looking at the differences. With this in mind, the proposed methods are applied to the experimentally collected condition data to diagnose the imbalance condition in a motor system.

Application of machine condition data to deep learning models

In this section, the deep learning models described in "Deep learning models using LSTM and CNN" section are trained and tested using the acceleration signals obtained from the experiment. As mentioned, the goal of paper is to develop a fault detection model whose accuracy is invariant to changes in the RPM. However, the models' performance at a constant RPM setting is evaluated first to show whether trained DL models are able to detect a targeted fault properly at the constant operating condition ("Constant RPM setting" section). Subsequently, a model's performance to data obtained from operating conditions different from the training data is studied ("Varying RPM setting" section).

Constant RPM setting

Before training the models, raw acceleration signals obtained from the experiment are divided into training (70%) and testing dataset (30%), and processed as described in "Data preprocessing for vibration signal" section. Because a constant RPM setting is considered here, the noise-reducing data transformation, i.e., scaling and smoothing, which described in "Scaling and smoothing of time-series data obtained from different RPM settings", is not included in this section. Instead, raw time-series data are used to train and test the TS-LSTM model. Time–frequency data are extracted using CWT, and frequencies between 0 and 400 Hz are used because the motor ran at low speeds during the experiment.

Once time-series and time-frequency data are prepared, training and testing are conducted in a PC platform (Precision 5820 Tower). To implement the proposed method, for hardware, any standard PC with a decent Nvidia GPU (preferable) will be enough because the deep learning models used in this study are relatively shallow, i.e., there are not many learnable parameters compared to a model often used for image recognition. For software, Python 3 along with some packages (e.g., numpy, PyTorch, and torchvision) are used. The PC platform used for this study is equipped with Intel Xeon with 32 GB RAM and GeForce GTX 1080 TI with 11 GB GDDR5X. The proposed deep learning architectures (TS-LSTM and CWT-CNN) are implemented through Pytorch deep learning framework.

Each model is trained with the Adam optimizer (Kingma and Ba 2015) over 500 epochs with a learning rate of 0.001 and a batch size of 16. The parameters were selected by a trial and error experiment using a technique called GridSearch, in which (1) sets of possible hyper-parameter values were taken, e.g., learning rates of 0.0001, 0.001, and 0.01 and batch size of 8, 16, 32, and 64, (2) every combination was tried once, and (3) the one that performed the best was chosen. To avoid

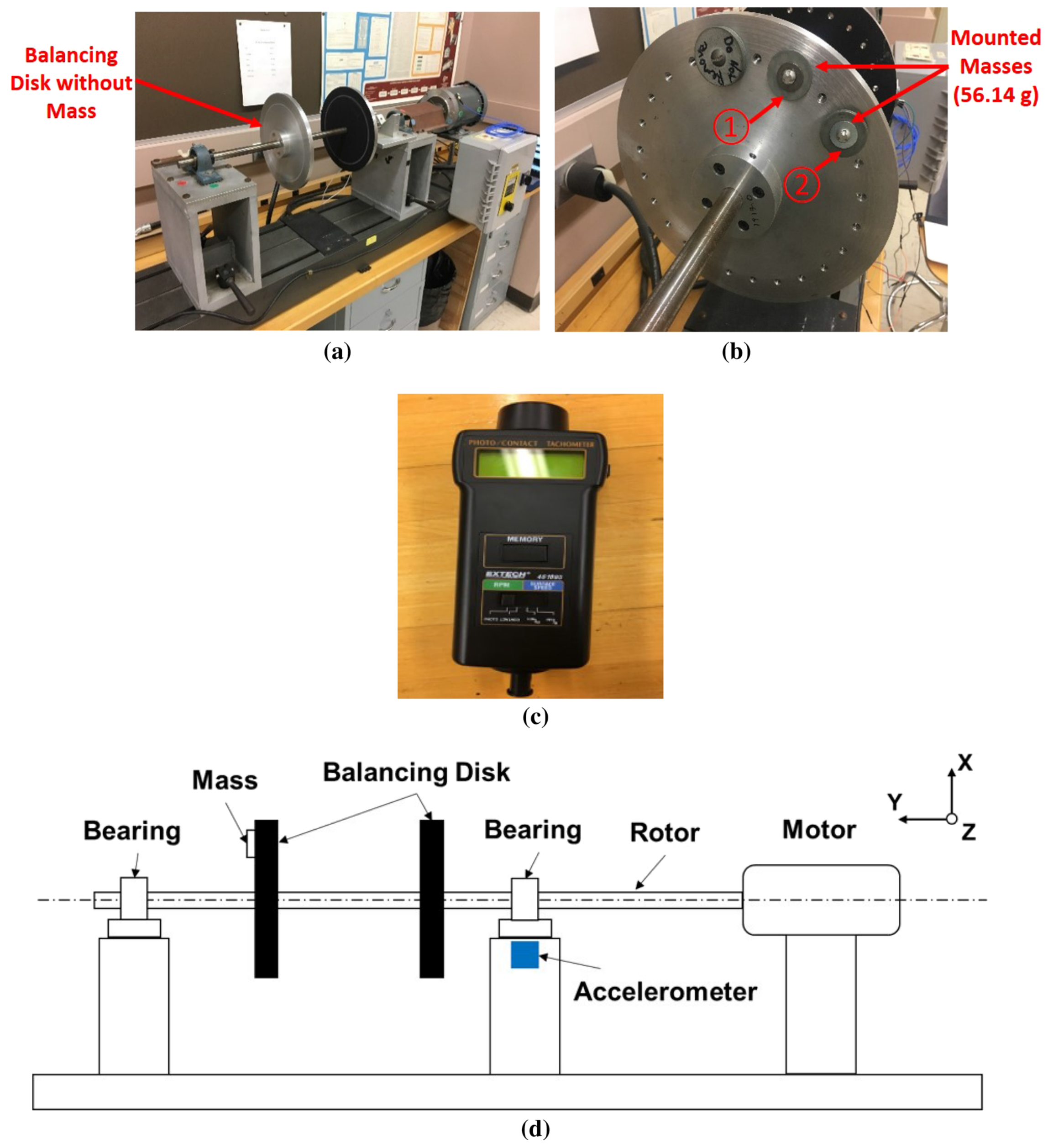


Fig. 5 Motor testbed for mechanical imbalance experiment; **a** overall configuration, **b** balancing disk with mounted masses, **c** tachometer, and **d** schematic diagram

an overfitting problem in the models, L2 regularization with lambda = 0.0002 is used, and models are trained to minimize the cross-entropy loss function (L),

$$L(y, \hat{f}_\theta(x)) = -(y \log(\hat{f}_\theta(x)) + (1-y)\log(1-\hat{f}_\theta(x)) + \lambda\|\theta\|^2, \tag{16}$$

where $\hat{f}_\theta(x)$ is the model output for input data, x, parameterized by weights θ, y is the true label of x, and λ is the L2 regularization weight. Each experiment include 25 trials and accuracies are reported with 95% confidence intervals.

Here, for constant RPM, a model is trained and tested using data collected from same RPM. Thus, both TS-LSTM and CWT-CNN architectures are used to develop models for each RPM settings, i.e., a model trained with data obtained at the rotational speed of 300, 320, 340, 360, or 380 RPM is tested with the data obtained at the rotational speed of 300, 320, 340, 360, or 380 RPM, respectively. Figure 7 shows prediction accuracies with 95% confidential intervals for TS-LSTM and CWT-CNN models (each model was trained and tested 25 times). As shown in the figure, CWT-CNN models outperform TS-LSTM models for

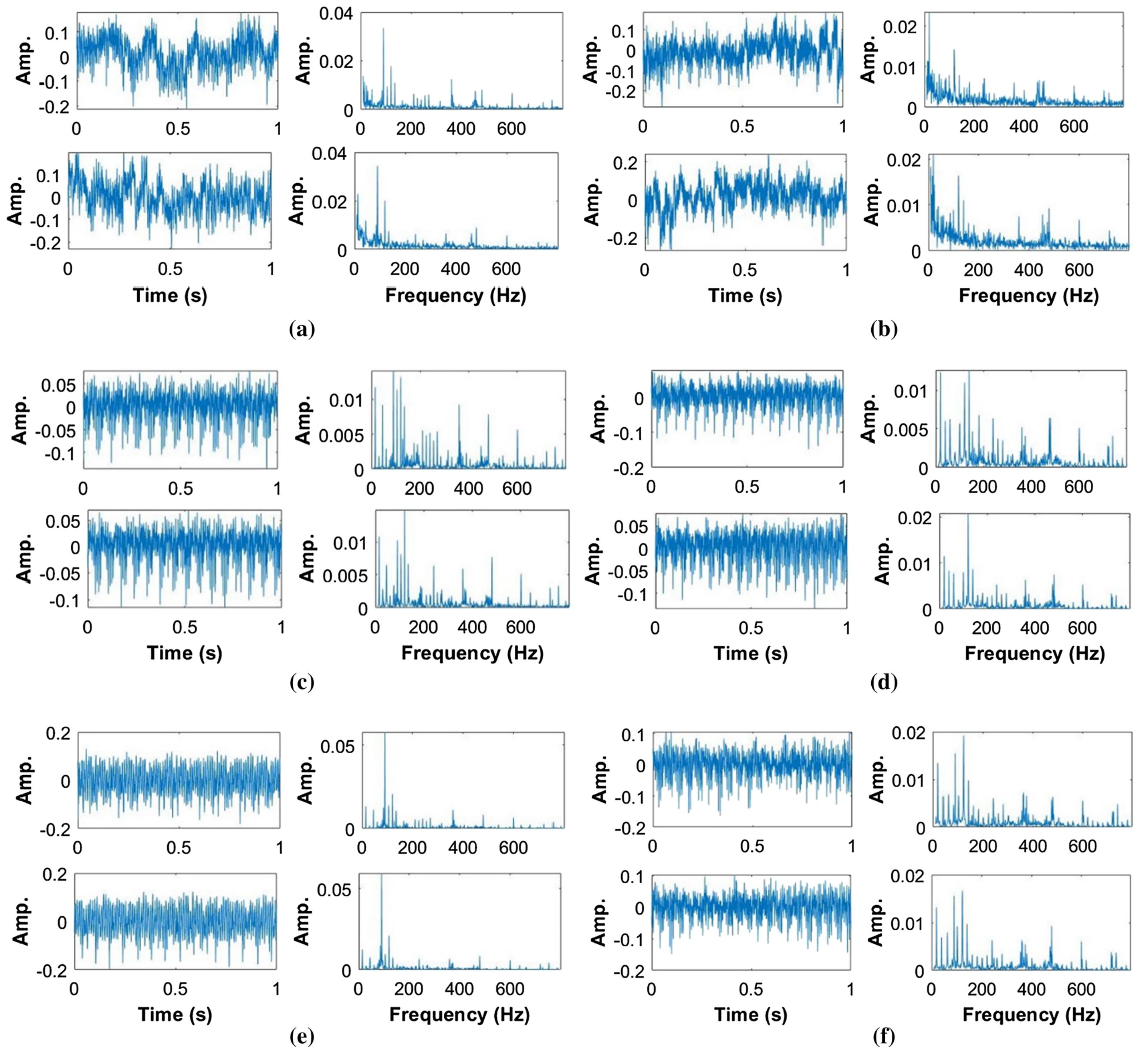


Fig. 6 Typical triaxial acceleration signals and their spectra for different imbalance levels (top is the default setting and bottom is the mass-loaded setting) and RPMs: **a** X axis and RPM = 300, **b** X axis and RPM = 380, **c** Y axis and RPM = 380, **d** Y axis and RPM = 380, **e** Z axis and RPM = 300, **f** Z axis and RPM = 380

all RPM cases. This is expected because raw signals were used in TS-LSTM models. Normally, acceleration signals are acquired with redundant information, which may not relevant to a machine condition. Due to the high accuracy of the CWT-CNN models, one may assume a possibility of overfitting. However, there is not much overfitting because all accuracies displayed in Fig. 7 are "test accuracies" (i.e., the test dataset was not used for training the models). Also, a poor test performance is normally observed in the case of overfitting. One possible reason for the high accuracy is because, on same-RPM training and testing, the differences between the machine condition data of the two states are so vast that it is fairly simple for the models to find a comfortable decision boundary.

In next section, the noise-reducing data transformation is implemented for the time-series data. A model is trained with data obtained from one RPM setting, and the trained model is evaluated using data obtained at previously unseen RPM settings to reflect varying RPM condition.

Table 1 Comparison of tri-axial acceleration signals using RMS (time-domain) and Kurtosis (frequency-domain) features

Axis	Feature	300 RPM Default setting	300 RPM Mass-loaded setting	380 RPM Default setting	380 RPM Mass-loaded setting
X	RMS	0.0609	0.0662	0.0621	0.0716
	Kurtosis	114.0695	68.8551	33.0453	25.7101
Y	RMS	0.0286	0.0258	0.0289	0.0304
	Kurtosis	59.7082	78.0904	41.52	91.8306
Z	RMS	0.0517	0.0518	0.0351	0.0352
	Kurtosis	400.8934	427.1034	64.1671	71.3961

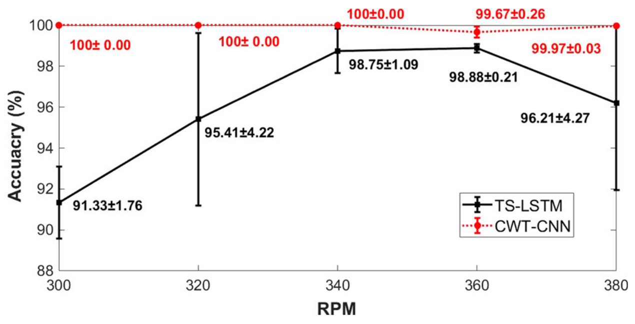


Fig. 7 Prediction accuracies with 95% confidential intervals for constant RPM settings

Varying RPM setting

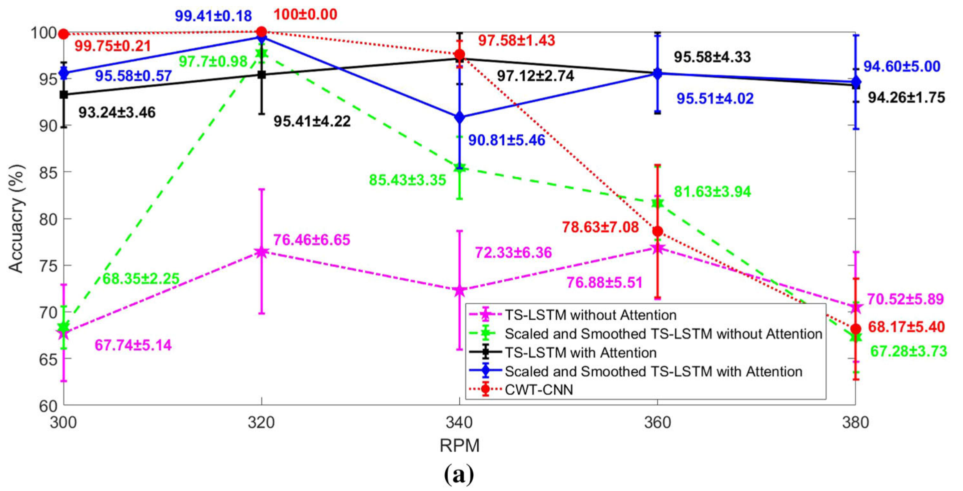
For the study of RPM invariance in a deep learning model, the LSTM and the CNN are trained with one dataset (i.e., data collected from one RPM setting), and tested with the other dataset (i.e., data collected at the other RPM setting). In this way, a model's performance variation with data from previously unseen RPM can be evaluated. Also, in order to demonstrate the effectiveness of the attention mechanism and the noise-reducing data transformation in the LSTM model, the performance of the LSTM model (1) without attention mechanism and (2) without noise-reducing data transformation are reported together.

Before training the models, the time series data go through the noise-reducing data transformation to extract features, which may have the property of RPM invariance in a LSTM model. To implement this, first, time-series data are scaled using α as described in "Scaling and smoothing of time-series data obtained from different RPM settings". Second, high frequency components are removed through a low pass filter because the motor ran at low RPMs. Less significant amplitudes, i.e., lower amplitudes, in the frequency domain are subsequently removed. Lastly, the filtered data in the frequency domain are converted back to the time domain using inverse DFT (per the procedure visualized in Fig. 1), and they are used to train and test the LSTM architecture described in "Long short-term memory (LSTM)" section (we call this as "Scaled and Smoothed TS-LSTM"). Time–frequency data are obtained through CWT as described in "Extracting time–frequency features using continuous wavelet transform" and "Constant RPM setting" sections, and the data are applied to CNN architecture explained in "Convolutional neural network (CNN)" section.

Trainings of the models were conducted on the same PC platform using the same parameters as described in "Constant RPM setting" section. A model was trained using dataset from one RPM setting (training RPM) and evaluated by the dataset (testing RPM), which were obtained from different RPM settings with the training dataset (i.e., training dataset and testing dataset were collected in different rotational speed settings). Here, five models, (1) TS-LSTM without Attention (2) Scaled and Smoothed TS-LSTM without Attention, (3) TS-LSTM with Attention, (4) Scaled and Smoothed TS-LSTM with Attention (proposed method), and (5) CWT-CNN were examined. Results of TS-LSTM and Scaled and Smoothed TS-LSTM can be used to demonstrate the effectiveness of the noise-reducing data transformation in the LSTM model. Similarly, result of the models with attention and without attention can be used to evaluate the effectiveness attention mechanism in the LSTM model.

To examine the performance variation of the five models, the models trained with data obtained from rotational speeds of (a) 320 RPM and (b) 340 RPM are selected, and prediction accuracies are displayed in Fig. 8 with 95% confidence intervals. In the figure, the performances of multiple models are plotted together to (1) graphically display the accuracy drift away as the new speed deviates from what has been tuned, and (2) compare the proposed method, i.e., Scaled and Smoothed TS-LSTM with Attention, with other typical methods. As described in "Invariance to changing rotational speed in fault detection" section, the goal of this study is to develop a method, which will be likely to have the property of RPM invariance using RPM transformation and deep learning model.

For CWT-CNN, in all experiments, the model's performance tends to drop significantly as the test data RPM differs from the training data RPM despite performing well for constant-RPM cases. This is expected as there was no RPM-invariant method implemented in the CNN model.



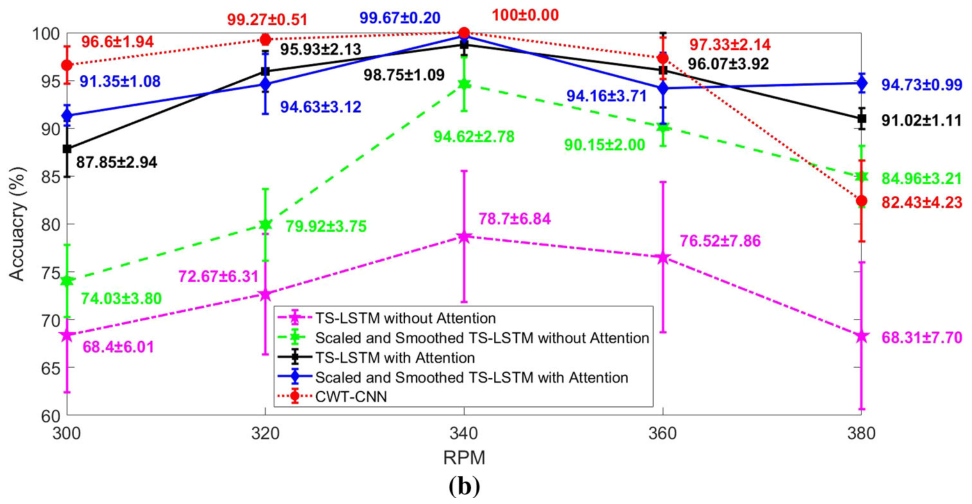


Fig. 8 Models' performance variation when considering data obtained from previously unseen RPM settings; a model was trained with data from **a** 320 RPM and **b** 340 RPM settings

Similarly, the LSTM models without attention mechanism also show this significant drop, but the model incorporating noise-reducing data transformation (i.e., Scaled and Smooth TS-LSTM) displays better performance than the model using raw data (i.e., TS-LSTM) in the most cases. The LSTM models, which have attention mechanism, do not show the significant drop in performance, especially in the Scaled and Smoothed TS-LSTM with Attention. The models with attention mechanism also can still often maintain accuracies above 90% from test data with significantly different RPMs. So, the effectiveness of the Attention mechanism in the LSTM model and the effectiveness of the scaling and smoothing of time-series method for the property of RPM invariance are demonstrated by comparing with other methods (or other combination).

In Table 2, the performances of the five models for the experiments, which are not included in Fig. 8, are summarized. In the table, the values in the boxes with italics present the performance for constant RPM setting. While the TS-LSTM with scaled and smoothed data does not outperform the TS-LSTM with raw data in every case, it performs better on average, especially in cases where the RPM difference is greater. This shows that the scaling and smoothing procedure on LSTMs does indeed provide significant benefits in varying RPM situations. Also, the prediction accuracies of the LSTM model are significantly improved by adding the attention mechanism in the model.

Conclusion

This paper has proposed a DL-based method for condition monitoring of rotating machinery that is invariant to changes to rotational speed. To experimentally validate the RPM invariance in a deep learning model, sets of experiments were conducted to collect machine condition data (i.e., triaxial acceleration) at various RPMs. The condition data were processed to extract features, which may better represent the RPM invariance in a model, and were applied to train and test the proposed LSTM and CNN architectures. The RPM invariance for the models was examined by using the data obtained from previously unseen RPM settings.

Table 2 Performances of the models when considering data obtained from previously seen (italics shading) and unseen RPM settings

Training RPM	Model	Testing RPM				
		300	320	340	360	380
300	TS-LSTM without attention	*98.66 ± 0.32%*	91.20 ± 2.15%	74.55 ± 5.74%	76.47 ± 5.48%	67.33 ± 5.71%
	Scaled and smoothed TS-LSTM without attention	*97.88 ± 0.47%*	79.38 ± 3.04%	82.60 ± 3.38%	83.95 ± 2.73%	59.06 ± 3.18%
	TS-LSTM with attention	*91.33 ± 1.76%*	77.66 ± 4.57%	64.75 ± 5.63%	65.87 ± 5.87%	58.67 ± 6.40%
	Scaled and smoothed TS-LSTM with attention	*98.50 ± 0.43%*	72.50 ± 3.51%	83.90 ± 3.85%	83.38 ± 3.92%	72.76 ± 4.65%
	CWT-CNN	*100 ± 0.00%*	65.17 ± 5.23%	58.17 ± 5.59%	51.50 ± 1.85%	50.63 ± 0.50%
360	TS-LSTM without Attention	77.18 ± 4.24%	85.25 ± 4.01%	86.37 ± 4.93%	*81.93 ± 6.12%*	81.12 ± 5.25%
	Scaled and smoothed TS-LSTM without attention	76.30 ± 4.17%	90.20 ± 3.02%	87.28 ± 3.77%	*95.09 ± 2.06%*	80.44 ± 4.71%
	TS-LSTM with attention	84.25 ± 4.44%	91.06 ± 2.56%	97.93 ± 0.63%	*98.88 ± 0.21%*	89.37 ± 0.99%
	Scaled and smoothed TS-LSTM with Attention	84.24 ± 4.83%	94.39 ± 1.39%	88.59 ± 4.99%	*98.62 ± 0.35%*	85.12 ± 6.04%
	CWT-CNN	96.6 ± 1.94%	99.27 ± 0.51%	97.33 ± 2.14%	*99.67 ± 0.26%*	84.43 ± 4.23%
380	TS-LSTM without Attention	52.91 ± 4.70%	55.38 ± 6.37%	72.15 ± 6.61%	64.29 ± 5.72%	*68.26 ± 7.26%*
	Scaled and smoothed TS-LSTM without Attention	70.14 ± 3.01%	83.87 ± 3.17%	76.50 ± 4.50%	73.54 ± 5.92%	*96.21 ± 2.06%*
	TS-LSTM with Attention	75.66 ± 5.07%	83.32 ± 5.14%	85.84 ± 6.05%	90.37 ± 5.72%	*96.21 ± 4.27%*
	Scaled and smoothed TS-LSTM with Attention	77.00 ± 3.58%	91.37 ± 4.87%	93.43 ± 2.59%	83.40 ± 4.55%	*98.15 ± 2.70%*
	CWT-CNN	61.05 ± 6.23%	87.75 ± 5.55%	96.93 ± 1.66%	99.83 ± 0.08%	*99.97 ± 0.03%*

Through the results of the DL experiment, a condition was well classified when tested on data with the same RPM as its training set. However, for all models, the prediction accuracies tended to degrade as the test data RPM began to differ. Overall, LSTM model showed smaller performance degradation to previously unseen RPMs than did the CNN model. Also, LSTM classifies the condition more effectively in terms of average accuracies. The Scaled and Smoothed TS-LSTM, in which raw signals are processed through the noise-reducing data transformation, is proven to be a better method than the TS-LSTM. From this, it is shown that the signal processing method can enhance the model's performance in the LSTM model. Also, the effectiveness of the attention mechanism in the LSTM model is demonstrated by comparing the performance of the models with and without attention mechanism.

In conclusion, this paper introduced RPM invariance, and it was tested through the proposed methods. Also, the models' uncertainties to varying speeds were quantified and compared. For a real world application, a condition monitoring system must identify a targeted fault under variable operational conditions. Thus, a model's invariance to varying operating condition must be considered in the diagnosis and prognosis of machine health. As a future work, a method which can detect a targeted fault under any speeds (i.e., when test RPM is unknown) will be studied.

Acknowledgements This work is supported by the Wabash Heartland Innovation Network (WHIN). Any opinions, findings, conclusions, and/or recommendations expressed are those of the authors and do not necessarily reflect the view of the WHIN.

References

Cacciola, S., Agud, I. M., & Bottasso, C. L. (2016). Detection of rotor imbalance, including root cause, severity and location. *Journal of Physics: Conference Series*. https://doi.org/10.1088/1742-6596/753/7/072003.

CESMII. (n.d.). https://www.cesmii.org/cesmii-announces-first-rfp-project-selections. Retrieved July 28, 2019.

Colah. (n.d.). https://colah.github.io/posts/2015-08-Understanding-LSTMs/. Retrieved August 12, 2019.

Janssens, O., Slavkovikj, V., Vervisch, B., Stockman, K., Loccufier, M., Verstockt, S., et al. (2016). Convolutional neural network based fault detection for rotating machinery. *Journal of Sound and Vibration, 377,* 331–345. https://doi.org/10.1016/j.jsv.2016.05.027.

Jia, F., Lei, Y., Guo, L., Lin, J., & Xing, S. (2018). A neural network constructed by deep learning technique and its application to intelligent fault diagnosis of machines. *Neurocomputing, 272,* 619–628. https://doi.org/10.1016/j.neucom.2017.07.032.

Jia, F., Lei, Y., Lin, J., Zhou, X., & Lu, N. (2016). Deep neural networks: A promising tool for fault characteristic mining and intelligent diagnosis of rotating machinery with massive data. *Mechanical Systems and Signal Processing, 72–73,* 303–315. https://doi.org/10.1016/j.ymssp.2015.10.025.

Jing, L., Zhao, M., Li, P., & Xu, X. (2017). A convolutional neural network based feature learning and fault diagnosis method for the condition monitoring of gearbox. *Measurement: Journal of the International Measurement Confederation, 111,* 1–10. https://doi.org/10.1016/j.measurement.2017.07.017.

Khan, S., & Yairi, T. (2018). A review on the application of deep learning in system health management. *Mechanical Systems and Signal Processing, 107,* 241–265. https://doi.org/10.1016/j.ymssp.2017.11.024.

Kim, D. B. (2019). An approach for composing predictive models from disparate knowledge sources in smart manufacturing environments. *Journal of Intelligent Manufacturing, 30*(4), 1999–2012. https://doi.org/10.1007/s10845-017-1366-7.

Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. In *International conference on learning representations* (pp. 1–15). http://arxiv.org/abs/1412.6980.

Lee, W. J., Mendis, G. P., Triebe, M., & Sutherland, J. (2019b). Monitoring of a machining process using kernel principal component analysis and kernel density estimation. *Journal of Intelligent Manufacturing*. https://doi.org/10.1007/s10845-019-01504-w.

Lee, W. J., Wu, H., Yun, H., Kim, H., Jun, M. B. G., & Sutherland, J. W. (2019a). Predictive maintenance of machine tool systems using artificial intelligence techniques applied to machine condition data. In *Procedia CIRP* (Vol. 80, pp. 506–511). Elsevier B.V. https://doi.org/10.1016/j.procir.2018.12.019.

LSM - Purdue University. (n.d.). https://engineering.purdue.edu/LSM. Retrieved September 10, 2019.

NN SVG. (n.d.). http://alexlenail.me/NN-SVG/LeNet.html. Retrieved August 12, 2019.

Park, J., Hamadache, M., Ha, J. M., Kim, Y., Na, K., & Youn, B. D. (2019). A positive energy residual (PER) based planetary gear fault detection method under variable speed conditions. *Mechanical Systems and Signal Processing, 117,* 347–360. https://doi.org/10.1016/j.ymssp.2018.08.010.

Peng, Z. K., & Chu, F. L. (2004). *Application of the wavelet transform in machine condition monitoring and fault diagnostics: A review with bibliography, 18,* 199–221. https://doi.org/10.1016/S0888-3270(03)00075-X.

Peng, Y., Dong, M., & Zuo, M. J. (2010). Current status of machine prognostics in condition-based maintenance: A review. *International Journal of Advanced Manufacturing Technology, 50,* 297–313. https://doi.org/10.1007/s00170-009-2482-0.

Pimenov, D. Y., Bustillo, A., & Mikolajczyk, T. (2018). Artificial intelligence for automatic prediction of required surface roughness by monitoring wear on face mill teeth. *Journal of Intelligent Manufacturing, 29*(5), 1045–1061. https://doi.org/10.1007/s10845-017-1381-8.

PyTorch. (n.d.). https://pytorch.org/. Retrieved August 13, 2019.

Seevers, J. P., Johst, J., Weiß, T., Meschede, H., & Hesselbach, J. (2019). Automatic time series segmentation as the basis for unsupervised, non-intrusive load monitoring of machine tools. In *Procedia CIRP* (Vol. 81, pp. 695–700). Elsevier B.V. https://doi.org/10.1016/j.procir.2019.03.178.

Vaswani, A., Brain, G., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., et al. (2017). Attention is all you need. In *Advances in neural information processing systems* (pp. 5998–6008). Retrieved September 13, 2019, from http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf.

Verstraete, D., Ferrada, A., Droguett, E. L., Meruane, V., & Modarres, M. (2017). Deep learning enabled fault diagnosis using time-frequency image analysis of rolling element bearings. *Shock and Vibration, 2017,* 1–17. https://doi.org/10.1155/2017/5067651.

Yoo, Y., & Baek, J. G. (2018). A novel image feature for the remaining useful lifetime prediction of bearings based on continuous wavelet transform and convolutional neural network. *Applied Sciences, 8*(7), 1102. https://doi.org/10.3390/app8071102.

Zhang, Z., Wang, Y., & Wang, K. (2013). Fault diagnosis and prognosis using wavelet packet decomposition, Fourier transform and artificial neural network. *Journal of Intelligent Manufacturing, 24*(6), 1213–1227. https://doi.org/10.1007/s10845-012-0657-2.

Zhang, J., Wang, P., Yan, R., & Gao, R. X. (2018). Long short-term memory for machine remaining life prediction. *Journal of Manufacturing Systems, 48*(May), 78–86. https://doi.org/10.1016/j.jmsy.2018.05.011.

Zhao, R., Yan, R., Chen, Z., Mao, K., Wang, P., & Gao, R. X. (2019). Deep learning and its applications to machine health monitoring. *Mechanical Systems and Signal Processing, 115*(15), 213–237. https://doi.org/10.1016/j.ymssp.2018.05.050.