A stacked autoencoder neural network based automated feature extraction method for anomaly detection in on-line condition monitoring

Roy, Mohendra; Bose, Sumon Kumar; Kar, Bapi; Gopalakrishnan, Pradeep Kumar; Basu, Arindam


2018

http://hdl.handle.net/10220/49567

© 2018 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works. The published version is available at: https://doi.org/10.1109/SSCI.2018.8628810
A Stacked Autoencoder Neural Network based Automated Feature Extraction Method for Anomaly detection in On-line Condition Monitoring

Mohendra Roy  
School of Electrical and Electronic Engineering  
Nanyang Technological University  
Singapore 639798  
mohendra.roy@ntu.edu.sg

Bapi Kar  
School of Electrical and Electronic Engineering  
Nanyang Technological University  
Singapore 639798  
bapik@ntu.edu.sg

Arindam Basu*  
School of Electrical and Electronic Engineering  
Nanyang Technological University  
Singapore 639798  
arindam.basu@ntu.edu.sg

Sumon Kumar Bose  
School of Electrical and Electronic Engineering  
Nanyang Technological University  
Singapore 639798  
Bose0003@e.ntu.edu.sg

Pradeep Kumar Gopalakrishnan  
School of Electrical and Electronic Engineering  
Nanyang Technological University  
Singapore 639798  
pradeepgk@ntu.edu.sg

Abstract—Condition monitoring is one of the routine tasks in all major process industries. The mechanical parts such as a motor, gear, bearing are the major components of a process industry and any fault in them may cause a total shutdown of the whole process, which may result in serious losses. Therefore it is very crucial to predict any approaching defects before its occurrence. Several methods exist for this purpose and many research are being carried out for better and efficient models. However, most of them are based on the processing of raw sensor signals, which is tedious and expensive. Recently, there has been an increase in the feature based condition monitoring, where only the useful features are extracted from the raw signals and interpreted for the prediction of the fault. Most of these are handcrafted features, where these are manually obtained based on the nature of the raw data. This of course requires the prior knowledge of the nature of data and related processes. This limits the feature extraction process. However, recent development in the autoencoder based feature extraction method provides an alternative to the traditional handcrafted approaches; however, they have mostly been confined in the area of image and audio processing. In this work, we have developed an automated feature extraction method for on-line condition monitoring based on the stack of the traditional autoencoder and an on-line sequential extreme learning machine (OSELM) network. The performance of this method is comparable to that of the traditional feature extraction approaches. The method can achieve 100% detection accuracy for determining the bearing health states of NASA bearing dataset. The simple design of this method is promising for the easy hardware implementation of Internet of Things (IoT) based prognostics solutions.

Index Terms—Condition Monitoring, Feature Extraction, Autoencoder, OSELM, Handcrafted Feature, Neural Network, predictive maintenance.

I. INTRODUCTION

Condition Monitoring (CM) of machine health and maintenance are parts of the routine job in all major process industries. This is also an integral part of the evaluation of risk and asset management [1]. Specially, the on-line monitoring and diagnosis of machine health help in assessing the impact of aging parts on the reliability of the overall system. Further, this helps in several important aspects such as improving system safety, reducing processing and operation time, increasing system availability, maintenance planning, alerting the crew about the impending failure to take proactive measures and so on [2]. Recently, there has been an increase in on-line machine health monitoring research aiming for predictive maintenance and to extend equipments service life, as well as to develop on-board Integrated Systems Health Management (ISHM). This has been fuelled by the recent development in the machine learning research [3]–[7]. Many of them are possible due to the increase in the computation capability, efficient models as well as the availability of dataset. To further accelerate this type of research, the National Aeronautics and Space Administration...
from the test dataset with raw features. This extracted features from the trained
the training dataset with smoothed input data to learn to get the compressed
our proposed method, the network $C$ (input layer of OSELM), and used for on-line learning. After the convergence
the remaining samples.

This has not only helped in solving the curse
and time consuming.
combined with "trial and error" strategies, which are tedious
are handcrafted approaches and require domain knowledge
features [17], [18], and so on. All these
alization, data understanding, reduce the memory requirement,
which streamlines the training and utilization times, data visu-
problem is to reduce the dimension of the raw input signals,
solutions [10]. A popular approach to solving this kind of
edge computing and Internet of things (IoT) based diagnosis
unwanted interference and noise. The dimensionality of the
data has a direct impact on the accuracy as well as training
time of the artificial neural network based models, and it turns
out to be more crucial for the power critical applications like
edge computing and Internet of things (IoT) based diagnosis
solutions [10]. A popular approach to solving this kind of
problem is to reduce the dimension of the raw input signals,
which streamlines the training and utilization times, data visu-
alization, data understanding, reduce the memory requirement,
and energy required for computation etc. Several methods
are there for achieving this [11]. Most commonly used time
domain methods are, (a) root mean square (RMS) [12], (b)
Kurtosis [13], [14], (c) Skewness [15], [16], (d) Crest factor
[15], [16], (e) Peak-to-Peak [17], [18], and so on. All these
are handcrafted approaches and require domain knowledge
combined with “trial and error” strategies, which are tedious
and time consuming.

Recent progress in autoencoder based feature extraction
method has provided an alternative means for feature extract-

tion [19], [20]. This has not only helped in solving the curse
of dimensionality [21], but also capable of providing more
discriminating features as compared to the traditional feature
engineering approaches [22]. Moreover, the best features for
that task are directly “learned” from the data avoiding adhoc
trial and error strategies. In some cases, it outperforms the
traditional handcrafted features [23]. A considerable amount
of progress has been made in the area of automated image and
speech feature extraction [24]. As per our knowledge, no such
effort has been made towards machine condition monitoring.

In this paper, we demonstrate a stack of the traditional
autoencoder (TAE) and an On-line Sequential Extreme Learning
Machine (OSELM) for automated feature extraction and
condition monitoring of bearing health. The objective of this
work is to develop an automated feature extraction method
followed by an on-line condition monitoring system, which is
comparable to the traditional feature extraction approaches.
The detail of this method and results are described in the
following sections.

II. METHODS

A. Stack of artificial neural networks

For the automated feature extraction, we have used the
traditional autoencoder [25]–[27], which was trained to find
the compress representation of the input data. The hidden
layer output from this trained autoencoder was then used as
an input to the boundary type OSELM [28] (see Fig. 1) for
the inference of the bearing health.

The traditional autoencoder is an artificial neural network
that can be trained to produce a replica of its input to its output
[29], each having $d$ nodes. The reduced code layer (also known
as hidden layer) $h$, having $L$ nodes, captures the compressed
representation of the input. This compressed representation
can be used as extracted features. An autoencoder consists of two
parts; the encoder part takes the input $X \in \mathbb{R}^d$ and maps it to the
hidden layer $h \in \mathbb{R}^L$, where

$$h = \sigma(WX + b)$$

(1)

Here $\sigma$, $W$ and $b$ denote the activation function of the
(hidden) latent layer $h$, input weight matrix, and bias vector
respectively. The decoder part maps the latent representation
$h \in \mathbb{R}^L$ to the output $X' \in \mathbb{R}^d$ (for autoencoder $m = d$). Here
$X' = \gamma(W'h + b')$ where $X'$ is the reconstruction of the input
and $\gamma$, $W'$, $b'$ are the activation function, output weight matrix,
and bias vector respectively.

In this work, we have used a single hidden layer autoencoder
with $d = 4096$, $L = 5$ and $m = 4096$. We have used the
Rectified Linear Units (ReLU) as an activation function in
both encoder and decoder parts [30], [31] and the weights
were optimized using the Adam optimizer [32].

The network was first trained in off-line mode using the
training dataset for one epoch, keeping out the test dataset for
the on-line feature extraction. The trained system (only the
encoder part) was then used for the on-line feature extraction
for the raw samples (see Fig. 1). Here the samples from the
test dataset were fed into the input layer of the autoencoder
(see network $B$ in Fig. 1) in sequence to compute the hidden

![Fig. 1. Graphical illustration of the proposed stack arrangement of neural networks for the on-line feature extraction and condition monitoring. This stack arrangement consists of two types of neural networks. The first type is a traditional autoencoder (TAE) having the same number of nodes in input and output layers, i.e., of 4096 neurons and a single hidden layer with five neurons (see network $A$). The second type of network (network $C$) is an on-line sequential extreme learning machine (OSELM), having an input layer with 5 neurons, hidden layers with 10 neurons, and a single output node. In our proposed method, the network $A$ was first trained in off-line mode using the training dataset with smoothed input data to learn to get the compressed representation of the data. This trained network (network $B$) was then used (only the encoder part of network $A$) for the on-line extraction of the features from the test dataset with raw features. This extracted features from the trained encoder (i.e. output of network $B$) were then fed into the classifier network $C$ (input layer of OSELM), and used for on-line learning. After the convergence of network parameters in $C$, this trained model is then used for inference of the remaining samples.](image-url)
layer output (the hidden layer output of network \( B \) is the same to the input of network \( C \) in Fig. 1). In each instance, these hidden layer outputs were then fed to the input layer of the OSELM (see the input of network \( C \) in Fig. 1) for the on-line training of OSELM and subsequently for the inference, i.e., condition monitoring.

The OSELM engine employs an on-line version of Extreme Learning Machine (ELM), which is known for its fast and straightforward learning method [28], [33]. Typically an ELM based network consists of a single hidden layer feed forward network, where the input weights (\( W^* \)) and biases (\( b^* \)) are generated randomly from a continuous probability distribution and are kept intact without updating. The hidden layer outputs are computed as in equation 1. Only the output weights of the network are learned, in a single step resulting in faster convergence.

In batch mode, the output weight matrix \( \beta \) is determined as:

\[
\beta = H^T \left( \frac{1}{C} + HH^T \right)^{-1} Y
\]

where \( H, I, Y \) and \( C \) are the hidden layer output, identity matrix, target output and a hyper-parameter respectively. For the on-line version of ELM, \( \beta \) is updated sequentially as follows:

\[
\beta_n = \beta_{n-1} + P_n^{-1} (Y_n - H_n \beta_{n-1}) H_n^T
\]

where \( P_n = (P_{n-1} + H_n H_n^T) \) and for the first batch of samples, i.e., \( n = 1 \), \( P_{n-1} = P_0 = \left( \frac{1}{C} + H_0 H_0^T \right) \) [34].

Here we have used the OSELM as one class classifier for anomaly detection, in boundary mode [35], [36]. This means that the network was trained for only one particular class, with only healthy samples from the machine assuming health condition has not degraded at the beginning of the lifetime. The output of the hidden layer was mapped to only one output node (for example \( Y \) is mapped to a value of 1 for healthy samples). As a result, it required less number of multiplication and accumulation (MAC) operation compared to that of the reconstruction mode where the number of output neurons (\( m \)) is the same as input dimension \( d \). The input layer of the OSELM engine has the same dimension as that of the hidden layer of the traditional autoencoder (see Fig. 1), i.e., \( d = 5 \). The hidden layer of OSELM engine is having a dimension of \( L = 10 \) and a single output node, i.e., \( n = 1 \) (see network \( C \) in Fig. 1). We have trained this network in an on-line fashion with features that were extracted by the traditional autoencoder (i.e., the trained encoder part of the autoencoder, see Fig. 1). The initial output weight of OSELM engine (\( \beta_0 \)) was calculated using the first 10 samples in a batch. Then the \( \beta \) was updated in on-line manner for each sample until it converged. Thus the network was trained by the on-line extracted features. Since the bearings are considered to be in healthy state at the earlier stage of their life, therefore the network was trained only for the healthy class. The converged network was then used for the on-line inference for rest of the samples. In this mode, if the samples were from the unhealthy state of the bearing, then the output of the network deviates from its trained class value of \( Y = 1 \). Any deviation from this value that surpasses a well defined threshold is considered as anomaly or a faulty state of the machine being monitored.

Since the algorithm converges very fast [37], therefore we can train it using a few samples (average convergence length is of 582 samples). The convergence criteria is based on how small the change of \( \beta \) has become during the update (or training), i.e., \( \Delta \beta \) should be very small for a fair number of consecutive samples. In this work, we adopted \( \% \Delta \beta < T_c \) for a consecutive ten samples as a convergence criteria. Here \( T_c = 0.1\% \) is used as the termination value. The average convergence length for the dataset with auto extracted features is 582, which is about 5 times less to that of the average sample length of the dataset (the average sample length of the NASA dataset is 3154). For the same dataset, the average convergence length is 481 with handcrafted features. This shows that for the both type of feature extraction, the network get converged within the first (1/5)\(^{th} \) of the total samples of the dataset. Thus OSELM is suitable for the on-line training and subsequently for inference using the same dataset. From application point of view, a different model has to be learned for each machine since the placement of sensors and type of fault of each machine may be unique. Hence, it is obvious if healthy models are learned for each machine separately in an online manner on the deployed hardware. The choice of ELM as the classifier is motivated by both its fast convergence as well as availability of power efficient hardware [38], [39] for deployment.

B. Dataset Preparation for on-line feature extraction

In this work, we used NASA Bearing dataset [8], provided by the Center for Intelligent Maintenance Systems (IMS), University of Cincinnati. There are three types of datasets, each of them consisting of vibration signals from four different bearings. The dataset 1 contains vibration signals for all the four bearings which were acquired from the accelerometers for both X and Y axis. For the dataset 2 and 3, the vibration signals for all the four bearings were obtained only from the accelerometers along the X-axis. These vibration signals were recorded for the duration of 1 second, at an interval of 10 minutes, with a sampling rate of 20KHz. The description of each bearing dataset and their health status are summarized in table I.

The raw signals from the accelerometers carry significant amount of noise, which were reduced by taking the average of every consecutive five raw data samples. This averaging helps in eliminating the high-frequency noise as well as reduces the number of input features in every sample from 20480 to 20480/5 = 4096. This help in reducing the number of computations and also yields more stable weight model during the training phase. We have used these filtered signals for rest of this work, i.e., for the training of the network as well as for the evaluation of bearing health. For this purpose, we have arranged the whole NASA bearing dataset into two parts, as training and testing dataset using leave one out strategy [40].
TABLE I
DETAILED DESCRIPTION OF NASA BEARING DATASET [8]

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Bearing</th>
<th>No. of Samples</th>
<th>No. of raw features in each sample</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Bearing1</td>
<td>2156</td>
<td>20480</td>
<td>No defect</td>
</tr>
<tr>
<td></td>
<td>Bearing2</td>
<td>2156</td>
<td>20480</td>
<td>No defect</td>
</tr>
<tr>
<td></td>
<td>Bearing3</td>
<td>2156</td>
<td>20480</td>
<td>Defect in inner race</td>
</tr>
<tr>
<td></td>
<td>Bearing4</td>
<td>2156</td>
<td>20480</td>
<td>Roller element defect</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Bearing</th>
<th>No. of Samples</th>
<th>No. of raw features in each sample</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Bearing1</td>
<td>984</td>
<td>20480</td>
<td>Outer race failure</td>
</tr>
<tr>
<td></td>
<td>Bearing2</td>
<td>984</td>
<td>20480</td>
<td>No defect</td>
</tr>
<tr>
<td></td>
<td>Bearing3</td>
<td>984</td>
<td>20480</td>
<td>No defect</td>
</tr>
<tr>
<td></td>
<td>Bearing4</td>
<td>984</td>
<td>20480</td>
<td>No defect</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Bearing</th>
<th>No. of Samples</th>
<th>No. of raw features in each sample</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Bearing1</td>
<td>6324</td>
<td>20480</td>
<td>No defect</td>
</tr>
<tr>
<td></td>
<td>Bearing2</td>
<td>6324</td>
<td>20480</td>
<td>No defect</td>
</tr>
<tr>
<td></td>
<td>Bearing3</td>
<td>6324</td>
<td>20480</td>
<td>Outer race failure</td>
</tr>
<tr>
<td></td>
<td>Bearing4</td>
<td>6324</td>
<td>20480</td>
<td>No defect</td>
</tr>
</tbody>
</table>

Since the NASA bearing dataset has total 12 bearings (see table I), therefore we have arranged 12 sets of datasets. Each set contains 11 bearing datasets for training and the left out one for the testing. For example, to evaluate the condition of the bearing1 of dataset1, we have treated the Dataset1 Bearing1 as test dataset (i.e. for on-line testing) and rest of the remaining 11 bearing datasets as training dataset (i.e. in off-line training mode).

C. Preparation of dataset with handcrafted features

For the comparison of our proposed method, we have performed the same on-line inference using five handcrafted features that extracted from the raw signals. We have used the following five conventional time-domain methods to create the handcrafted features: RMS, kurtosis, skewness, crest factor and peak-to-peak. Thus we have the dataset of 12 bearings with five handcrafted features, to be applied directly to the OSELM engine represented by network C, as highlighted in Fig. 1. The method of training and inference of the OSELM engine remains the same as described in the previous section.

III. RESULTS AND DISCUSSION

To check the reliability of the proposed method, we obtained the maximum amount of deviation of OSELM output to that of the healthy state of the same bearing, for the entire lifetime of the bearing (i.e. length of the dataset). Fig. 2 (a) shows the maximum deviations for each of the bearing, using automatically extracted features from the autoencoder. From the figure we can see that the Dataset1 Bearing3, Dataset1 Bearing4, Dataset2 Bearing1, and Dataset3 Bearing3 are having comparatively large deviation than the rest of the bearings. This is because all these four datasets are from faulty bearings according to the ground truth (see the dataset description in table I). The results from the handcrafted features also follow a similar behavior (see Fig. 2(b)). These results also show that the proposed method is able to indicate the health status of the faulty bearings.

However, we want to develop an automated process that spontaneously indicate the condition of the bearing health. This requires a scheme to identify an appropriate threshold value. Therefore we have implemented an adaptive thresh-
training of the autoencoder to find an optimal weight and bias values in OSELM engine. Subsequently, we evaluated the parameters $\mu_t$, $\sigma_t$ and then found the value of $K$ which determined the true state of the corresponding bearing dataset. We selected the $K$ value that yields the maximum accuracy in determining the bearing health state of all the 11 bearings in the training dataset.

After finding the optimum value of $K$, we then used it for the actual test dataset, i.e., which was left out as a test dataset and evaluated its performance by determining the true state of the bearing health as per as the ground truth (as described in the table I). In all cases, the value of $K$ chosen from training data was able to successfully classify both the healthy as well as the faulty bearing during testing. Optimal $K$ values, that helps in detecting the health status of each test dataset correctly, are shown in Fig. 3. Similar $K$ values corresponding to the handcrafted features are also shown in black line in Fig. 3.

We further validate our findings by evaluating the percentage of accuracy in determining the bearing health state of all the 12 bearings against a range of $K$ values in inference mode of OSELM engine. Here the percentage of accuracy implies the accuracy in detecting the health of all the bearings in the context of ground truth as described in table I. The experimental results in Fig. 4 presents the variation of $K$ and the percentage of accuracy for the auto-extracted features. This plot shows that the accuracy reaches 100% for $K \approx 10$. This means that the predictions for all the bearings using $K \approx 10$ is analogous to the ground truth. Similar observation for handcrafted features shows that 100% accuracy can be achieved for $K \approx 25$. These values roughly match the average values of $K$ obtained from the training dataset as described earlier (see Fig. 3).

Although the respective $K$ values for the automated method and the handcrafted method are different, both the approaches can predict the health of all the bearings with 100% accuracy. From these results, it is evident that the automatically extracted features are as useful as handcrafted features in detecting the bearing health condition. Since the proposed automated feature extraction method does not require any prior knowledge of the dataset and it requires to train only once for a particular type of dataset, therefore the proposed method is more feasible for applying to different machines.

It is to be noted that the above results are valid for a particular seed value used for generating the random input weights and bias values in OSELM engine. Subsequently, we performed the same exercise for five epochs with different seed values, in order to validate the robustness of the proposed method. This shows an average accuracy of 98.4% with a standard deviation of 3.2%. Needless to say that the value of $K$ for each seed was adjusted using the same optimization procedure as described above.

IV. CONCLUSION

In this paper, we demonstrate an automated feature extraction method that can extract a compressed representation of the raw signals, which can be utilized for the on-line condition monitoring of machine health. The performance of this method is similar to that of the handcrafted (manual) feature extraction approaches. Once the training of the autoencoder is performed in off-line mode, it is ready for automatic extraction of the

![Fig. 3. Evaluation of $K$ for determining the adaptive threshold for each of the test dataset. Here the plot in black is for the dataset with traditional handcrafted features and the plot in magenta is for the dataset with the automatically extracted features. In both cases the $K$ value was evaluated using the leave one out strategy.](image)
features from raw data using the trained hyper-parameters. Given that, the proposed approach can be realized in hardware for in-situ feature extraction and subsequent classification of the data as desirable in low power Edge-computing platforms for IoT applications.

In future, we intend to study the performance of various classifiers using the same automatically extracted features and also an ELM based ensemble engine.

ACKNOWLEDGMENT

We would like to thank the reviewers for their valuable comments for the improvement of this paper.

REFERENCES


