# A fault detection framework based on LSTM autoencoder: a case study for Volvo bus data set*

Narjes Davari[1], Sepideh Pashami[2,7], Bruno Veloso[1,5,4], Sławomir Nowaczyk[2], Yuantao Fan[2], Pedro Mota Pereira[6], Rita P. Ribeiro[1,3], and João Gama[1,4]

[1] INESC TEC, Porto, Portugal
[2] Center for Applied Intelligent Systems Research, Halmstad University, Sweden
[3] Faculty of Sciences, University of Porto, Porto, Portugal
[4] School of Economics, University of Porto, Porto, Portugal
[5] University Portucalense, Porto, Portugal
[6] Metro of Porto, Porto, Portugal
[7] RISE research institute of Sweden, Kista, Sweden

**Abstract.** This study applies a data-driven anomaly detection framework based on a Long Short-Term Memory (LSTM) autoencoder network for several subsystems of a public transport bus. The proposed framework efficiently detects abnormal data, significantly reducing the false alarm rate compared to available alternatives. Using historical repair records, we demonstrate how detection of abnormal sequences in the signals can be used for predicting equipment failures. The deviations from normal operation patterns are detected by analysing the data collected from several on-board sensors (e.g., wet tank air pressure, engine speed, engine load) installed on the bus. The performance of LSTM autoencoder (LSTM-AE) is compared against the multi-layer autoencoder (mlAE) network in the same anomaly detection framework. The experimental results show that the performance indicators of the LSTM-AE network, in terms of F1 Score, Recall, and Precision, are better than those of the mlAE network.

**Keywords:** Fault detection, outliers, time series, LSTM, Autoencoder

## 1 Introduction

In many industries, maintenance is a significant part of the operation. As an example, a key parameter for bus operators is vehicle downtime, namely whenever a vehicle is needed but not available [17]. Analysing the time buses from a particular fleet spend in a workshop (or on the way there) is vital for understanding the efficiency of operations, especially for follow-up on any improvements. An essential next step is to develop systems that can automatically detect faults, i.e., analyse the data on-board vehicles and identify anomalous behaviour. A fault

* This work was supported by the CHIST-ERA grant CHIST-ERA-19-XAI-012, project CHIST-ERA/0004/2019 funded by FCT - Fundação para a Ciência e Tecnologia and project 2020-00767 funded by Swedish Research Council.

prevents the bus from operating. One of the reasons for the surprisingly large amount of (costly) time buses spend at workshops is the waiting time – even though there is no work being done on it. The bus operator cannot optimally plan their operation since much of this waiting time is insufficient planning. Unexpected failures typically lead to long waiting times. The framework proposed in this paper aims to support the automotive industry in more efficient planning.

Anomalies are good indicators of malfunctions in a system. In the era of big data, considerable research efforts focus on designing online algorithms capable of detecting anomalies from streaming data. To detect anomalies in a given system, it is necessary to define a "normal system behaviour". However, when the volumes of data and the complexity of systems are continuously growing, it becomes infeasible for human experts to build an exhaustive definition of each system's normal behaviour. Moreover, the definition of normal is dynamic, as sensors generate data that is subject to change over time due to external conditions (i.e., normal data samples are drawn from a non-stationary distribution). In a real-world application domain, we monitor one bus operating in typical conditions in Sweden. We are particularly interested in detecting deviations that identify faults during a bus's operation. In this paper, we implement a fault detection framework based on deep learning (DL) to detect failures of bus air system. Our goal is to identify abnormal behaviours in the data stream obtained from sensors installed in the system while the bus is in operation. The objective is to predict if a failure evolves using unsupervised methods based on deep learning.

The remainder of the paper is structured as follows: an overview of the related work in the context of anomaly detection is provided in Section 2. Section 3 discusses the problem description. In Section 4 we present fault detection methodology and proposed failure detection framework. The case study, pre-processing and data cleaning, feature generation, and anomaly detection are discussed in Section 5. Section 6 contains experimental results obtained by the LSTM autoencoder (LSTM-AE) and multi-layer autoencoder (mlAE), and finally, the concluding remarks are provided in Section 7.

## 2 Related Work

The current industrial solution for vehicle on-board fault detection and diagnostic systems, e.g., [15], still rely heavily on domain knowledge from a human expert and is essentially based on either building a pattern recognition classifier or a reference model. This paradigm requires domain experts such as field engineers to drive the development, i.e., modelling the physical process involved, determining potential faults or risky events, conducting controlled experiments, and collecting relevant data for analysis. Relevant reviews can be found in [8,7,6,18]. While this paradigm has proven effective for predefined faults by domain experts, unexpected faults occurred post-deployment in the field not covered by the system, which is developed prior to the deployment.

An alternative approach is to monitor systems on-board vehicles and autonomously captured key characteristics of the operation (in model space) for anomalous event detection: the reference system representation is learned with unsupervised models (e.g., LSTM or AE networks) on data stream coming from machines working under normal conditions. In contrast, an abnormal machine would yield a deviation in the model space and a higher reconstruction error. Similar concepts are studied in, e.g., [10,3].

Deep learning methods have been employed for many different real-world applications. Fan et al. [5] utilised echo state network to capture air system dynamics and perform conformal anomaly detection with learned features for detecting compressor faults. Munir et al. [13] presented a DL approach to detect a range of anomalies (point anomalies, contextual anomalies and discords) in time series data. Michau et al. [12] used AE network for unsupervised feature parameter learning and integrated it with a one-class classifier that is only trained with samples of healthy conditions for fault detection. Davari et al.[4] proposed a data-driven predictive maintenance framework for the air production unit system of a train by deep learning based on a sparse AE network that efficiently detects abnormal data and considerably reduces the false alarm rate.

The anomaly detection techniques for time series sequence based on DL algorithms augmented with LSTMs are used in several studies (e.g., [9,19]). Chauhan et al. [1] applied recurrent neural network (RNN) and LSTM to detect anomalies in ECG signals. Nguyen et al. [14] proposed a LSTM based method for forecasting multivariate time series data and an LSTM AE combined with a one-class support vector machine algorithm for detecting anomalies in sales. Maleki et al. [11] introduced a probability criterion based on the central limit theorem to evaluate the likelihood of a data point that is drawn from an unknown probability distribution for the goal of data labelling. Then, normal data is passed to train an LSTM autoencoder that distinguishes anomalies when the reconstruction error exceeds a threshold.

This paper proposes a framework based on LSTM autoencoder to address the challenges and limitations of anomaly detection. The contribution of this study is a multivariate time series anomaly detection method based on LSTM autoencoder with the application to data from a Volvo bus in regular operation.

## 3 Problem Description

***Data Description*** The data used in this study were collected from buses operated in traffic around a city on the west coast of Sweden. Four vehicles were year model 2009, one was 2008, and the remaining was produced in 2007. On-board data collection took place from August 2011 until the end of 2017, in regular operation, where each bus was driven approximately 100 000 km per year.

Data from the J1587 diagnostic bus and two CAN buses (the vehicle and the powertrain CANs) were sampled once per second, collecting approximately one hundred sensor and control signal values. An in-house developed system called

the Volvo Analysis and Communication Tool (VACT) was used in the study; it uses a telematics gateway for communication and can wirelessly receive new sampling configurations.

In addition, we have analysed an off-board database containing Vehicle Service Record (VSR) information. Each entry in this database contains information about repair and maintenance services done on the vehicles, including date, mileage and unique part identifiers. There are, unfortunately, frequent quality issues with the data since the VSR is primarily manually entered by maintenance personnel. Given that the primary purpose is accounting and invoicing, the detailed information about the repairs, especially the degree of component deterioration and root cause analysis, is less than perfect. This data was partly curated using vehicle GPS data and bus operator's internal operation notebooks.

***Fault detection*** The key to reducing downtime is building a system capable of detecting early symptoms of wear and faults. If the operator and workshop personnel become alerted before they become real problems or failures, i.e., before they take the bus out of commission, they can be handled much more efficiently. Optimally, one could solve these problems during the next planned maintenance visit. In our study, we have noticed that vehicles would spend, on average, almost 1.5 months per year in workshops. Early discovery of faults and improved diagnostics is expected to decrease the waiting time, incorrect repairs significantly, and other similar issues, conservatively reducing the total downtime by 50% or more.

In this study, we aim to detect periods of abnormal vehicle operation, i.e., quantify the "strangeness" of sensor data, compared to what is expected. Many current approaches for equipment monitoring require (semi-)manual creation of some model of what is expected. On the other hand, our goal is to automatically monitor a wide range of complex equipment with many possible faults. This goal requires autonomously constructed knowledge from the data, with very little reliance on human experts. In particular, one cannot assume that a list of all possible faults can be provided for training.

## 4 Fault Detection Methodology

This experimental fault detection framework aims to predict and detect faults by cleaning and extracting time series data features in an optimal sliding time window and feeding them into a deep LSTM-AE network that performs a classification task.

***LSTM Encoder-Decoder*** An autoencoder is an unsupervised neural network (NN) trained to reconstruct the inputted time-series data as its output. The encoder learns to compress a high-dimensional input to a low-dimensional latent space, and the decoder then attempts to reconstruct the output with minimal error faithfully. The general form of multivariate time-series at the sliding time window $i$ can be expressed as $\mathbf{X}^{(i)} = \mathbf{X}_1^{(i)}, \mathbf{X}_2^{(i)}, \ldots, \mathbf{X}_j^{(i)}, .., \mathbf{X}_m^{(i)}$ of length $m$,

where $m$ is the number of time series variables (number of sensors) and $\mathbf{X}_j^{(i)}$ is an observation vector of readings from $j^{th}$ sensor at the sliding time window $i$.

The difference between the input vector $\mathbf{X}^{(i)}$ and the reconstructed vector $\hat{\mathbf{X}}^{(i)}$ is called the reconstruction error $\mathbf{e}_r^{(i)}$. The trained network tries to minimise the reconstruction error as its objective function. A common metric for this error is Mean Squared Error (MSE): $\mathbf{e}_r^{(i)} = \|\mathbf{X}^{(i)} - \hat{\mathbf{X}}^{(i)}\|^2$ which measures the proximity of the reconstructed input to the original input.

The trained LSTM Encoder-Decoder model reconstructs the normal multivariate time series. The reconstruction errors of the training data are then compared to test data, i.e., an anomaly score for each dataset in a sliding time window is calculated, and identifies whether it follows the normal distribution of the time series. The higher the anomaly score, the more likely is it that the given data time window should be considered an anomaly.

Figure 1 illustrates the steps of the LSTM-AE network for a time-series data consisting of $n$ sliding time windows, in which $\mathbf{h}_i^E$ and $\mathbf{h}_i^D$ are the hidden state of the encoder and decoder, respectively, at the sliding time windows $i = 1, \ldots, n$.

The LSTM encoder learns an input time series and generates an encoded state while the LSTM decoder produces the reconstructed data at the sliding time window $i$ by applying the hidden decoding state at sliding time window $i$ and the predicted time series at the sliding time window $(i-1)$. In order to reconstruct the time series, the encoder and decoder parts are jointly trained.

In order to obtain the hidden state of the encoder at sliding time window $i$, $\mathbf{X}^{(i)}$ at sliding time window $i$ and the hidden state of the encoder at sliding time window $(i-1)$, ( $\mathbf{h}_{i-1}^E$) are used. The hidden state of the encoder at the end of the input sequence, $\mathbf{h}_n^E$, is used as the initial state of the decoder, $\mathbf{h}_n^E = \mathbf{h}_n^D$. The decoder uses hidden state $\mathbf{h}_i^D$ and the predicted value of time series at sliding time window $i$, (i.e., $\hat{\mathbf{X}}^{(i)}$) to produce the next hidden state.

***The Proposed Framework*** The time-series dataset includes normal and abnormal observations. We split the normal data into two sets: training and validation. The training dataset is used to learn the LSTM-AE network, while the validation dataset is used for an early stop in the autoencoder training (i.e., when the validation loss does not improve and the generalisation error begins to degrade). The root mean square of reconstruction error for the training dataset ($RMSE_r^{train}$) is used to estimate a threshold value (through a Boxplot analysis) for labelling test data. The training dataset is assumed to have normal behaviour, so as we can consider, the $RMSE_r^{train}$ follows a normal distribution. However, if there are some outliers in the training dataset, the distribution will be asymmetric; i.e., the methods work based on normality assumption may not be useful. Boxplot is useful as a consistent method to display the distribution of the dataset. Extreme observations can be easily ignored, thus, it can be used to set the threshold of $RMSE$ of test data ($RMSE_r^{test}$).

Next, the test data that contain normal and abnormal samples is employed to validate the network performance. If $RMSE_r^{test}$ is larger than the threshold value, then the data is considered as an anomalous observation; other-

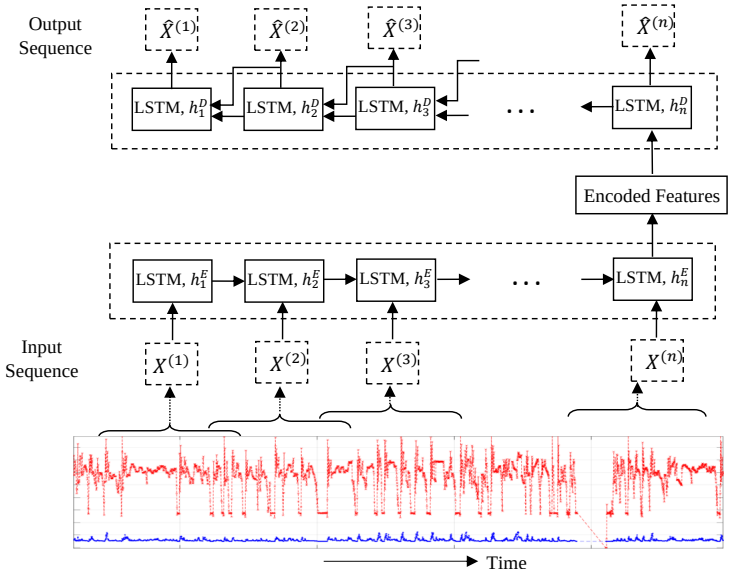


**Fig. 1.** An LSTM-AE network for a sequence with length $n$

wise, it is a normal one. In this paper, the anomalies are detected as a classification problem, where the classification labels "0" and "1" indicate normal and anomaly observations, respectively. The maximum and minimum values of the Boxplot are obtained from $1.5 * IQR$ above the third quartile ($Q_3$) and $1.5 * IQR$ below the first quartile ($Q_1$), respectively; where $IQR$ is the interquartile range, i.e. the difference between the upper and the lower quartiles. The interval $[Q1 + 1.5 * IQR,\ Q3 + 1.5 * IQR]$ contains 99.3% of data. Therefore, points outside this interval are considered as an anomaly [16].

Finally, the above output is post-processed using a low-pass filter through which the sudden variations are removed, decreasing the number of false alarms [16]. The flow chart of the framework is shown in Figure 2.

## 5 Case Study

A data-driven fault detection framework is developed that issues an alert whenever one of the key components in a specific bus exhibits an abnormal behaviour. The focus of the study is on the readings from ten sensors (e.g., wet tank air pressure, engine speed, engine load) installed on the bus by which real-time data was logged at 1 Hz frequency by an on-board embedded device. The anomaly detection framework performs data pre-processing, learns a network to combine several sensors readings, and identifies anomalies in sensor readings that can be symptoms of imminent faults. In order to evaluate the performance of the proposed framework, we compare the alarms raised by the framework against bus repair records.

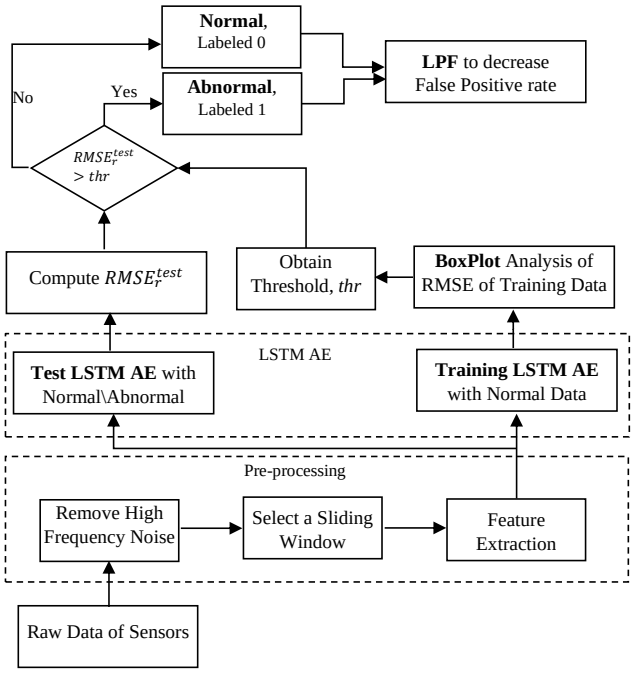


**Fig. 2.** Flow chart of fault detection framework.

***Analysis and cleaning of the Input Data.*** To reduce the influence of noisy data and outliers, we remove the high-frequency noises through a low pass filter (LPF) in pre-processing stage of data [2]. It encourages training data distribution to be close to the Gaussian distribution. In other words, the goal is to reconstruct the noisy data so that its distribution becomes similar to the normal distribution. Figure 3 shows the measured data of "engine speed" sensor during a short duration. The blue curve shows the raw data before filtering, and the data after filtering is shown in red. It is visible that the variations of raw data over time are very noisy, which can be further smoothed by the LPF.

Fast Fourier Transform (FFT) of the raw and filtered data on the frequency domain is shown in the right part of Figure 4. It shows that the raw data with frequencies less than 0.1 Hz have higher FFT values while the data with frequencies higher than 0.1 Hz have almost similar low FFT values indicating the specification of white noise. Figure 4 left shows the Probability Density Function (PDF) for the raw and the filtered engine speed data. The PDF of raw data (blue curve) shows a sideband peak around the main peak, which can be removed through filtering (red curve).

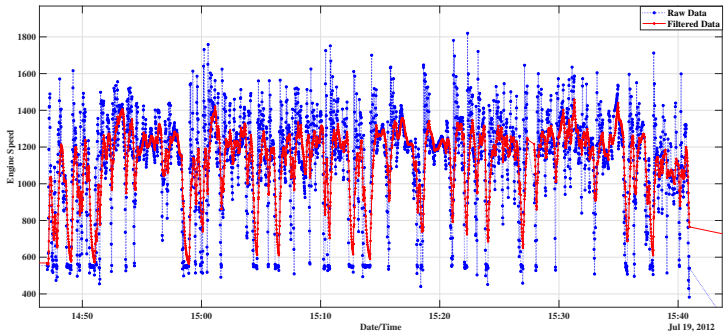


**Fig. 3.** The raw and filtered data of engine speed over time

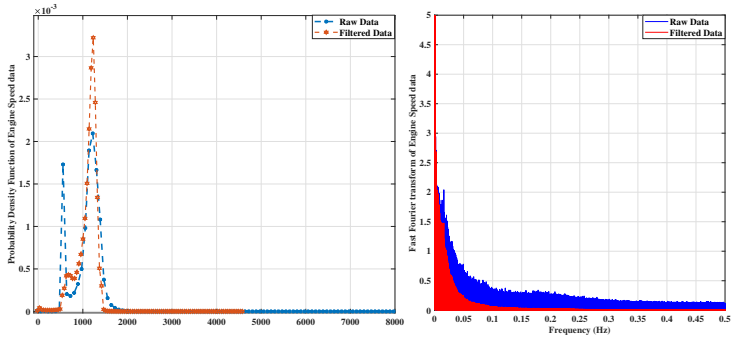


**Fig. 4.** The Probability Density Function - PDF (left) the Fast Fourier Transformation - FFT (right) for raw and filtered engine speed data.

***Feature extraction*** After cleaning, the raw data must be parsed to extract relevant information that could allow us to detect suspicious behaviours. Although the autoencoder reduces the dimension of multivariate data points to improve the network's performance, raw data with a large learning dimension cannot be directly applied. Therefore, statistical values of the multivariate time series in an selected sliding time window are used as input into the network. These statistical values (features) for $j^{th}$ sensor include mean, $x_j^{(1)}$, standard deviation, $x_j^{(2)}$, skewness, $x_j^{(3)}$, kurtosis, $x_j^{(4)}$, quantiles, $\{x_j^{(5)}, \ldots, x_j^{(8)}\}$, and deciles $\{x_j^{(9)}, .., x_j^{(18)}\}$, approximating the original raw data. Thus, through the pre-processing, for each sensor in a sliding time window 18 statistical features are computed; i.e., the dimension of the input to the network is 180 for 10 sensors. This conversion reduces the dimensionality and better expresses the characteristics of the data.

***Anomaly Detection*** As mentioned earlier in Section 4, through the Boxplot analysis, the distribution of all $RMSE_r^{train}$ is obtained, and its maximum value is set as a threshold ($thr$) with which normal and abnormal data in the test

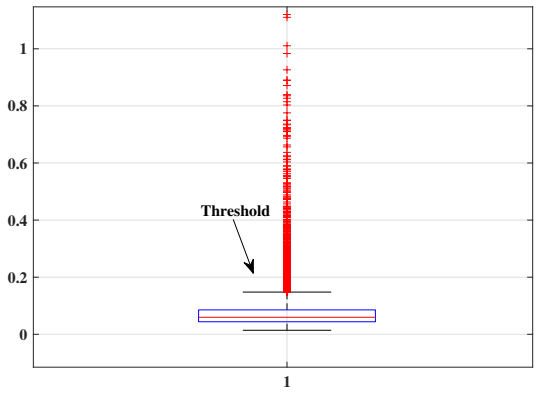


**Fig. 5.** The Boxplot analysis of the $RMSE$ of training dataset.

dataset are labelled as 0 and 1, respectively. Alternatively, the $RMSE$ could be computed using a separate validation dataset, however, we found it does not make a significant difference. Figure 5 shows the Boxplot of all $RMSE_r^{train}$. Note that $thr$ is calculated for the training dataset to identify abnormal behaviours of the test dataset, i.e., if $RMSE_r^{test} > thr$ then the respective sliding time window is detected as an anomaly.

## 6 Experimental results

This section presents and discusses the experimental results of the case study introduced in Section 5. We consider two different downtimes of the bus: unplanned and planned interruptions. The former occurs when one of the bus systems fails; i.e., the vehicle may be inoperable and towed to the workshop for repair, while the latter happens when the workshop personnel determines that a system is not functioning satisfactorily and decides to repair or replace it. A failure report provided by the company (see Table 3) describe repair information, particularly the date and operations performed and some comments. These off-board data makes it possible to evaluate true and false, positive and negative, alarms raised by the proposed anomaly detection framework. In order to evaluate the performance of the proposed solution, we report the following metrics: Recall, Precision, and F1 Score (%).

***Impact of Size of Sliding Window*** Here, we evaluate the impacts of different sizes (ranges from a smaller window size $1min$ to a larger window size of $10min$) of the sliding time window on the performance of the proposed framework. Note that the sliding time window with the same size is applied for the training and test datasets. Table 1 shows the metrics for the framework using $W_1$, $W_2$,.., $W_{10}$, i.e., sliding time windows with length $1min$, $2min$,.., $10min$, respectively. The

results shown (in bold) indicate that the sliding time window of size $4min$ ($W_4$) leads to the largest value of $TP$ and the lowest value of $FP$. Generally speaking, the value of $TP$ decreases with increasing the sliding time window length but the $FP$s increases alongside.

**Table 1.** Effect of sliding time window size in performance of LSTM-AE

| Metrics | W1 | W2 | W3 | W4 | W5 | W6 | W7 | W8 | W9 | W10 |
|---|---|---|---|---|---|---|---|---|---|---|
| $TP$ | 14 | 17 | 18 | **22** | 19 | 16 | 14 | 12 | 11 | 12 |
| $FP$ | 5 | 4 | 4 | **3** | 4 | 6 | 6 | 4 | 7 | 8 |
| $FN$ | 12 | 9 | 8 | **4** | 7 | 10 | 12 | 14 | 15 | 14 |
| $Recall(\%)$ | 53.8 | 65.4 | 69.2 | **84.6** | 73.1 | 61.5 | 53.8 | 46.1 | 42.3 | 46.1 |
| $Precision(\%)$ | 73.7 | 80.9 | 81.8 | **88.0** | 82.6 | 72.7 | 70.0 | 75.0 | 61.1 | 60.0 |
| $F1\ Score(\%)$ | 62.7 | 72.3 | 75.1 | **86.2** | 77.5 | 66.6 | 60.8 | 57.1 | 49.9 | 52.1 |

***Comparison of LSTM-AE and mlAE*** For evaluation purposes, we experimentally studied and tuned different LSTM-AE settings, and then the performance of the LSTM-AE was compared versus the mlAE. Since the network topology needs to be consistent with the experimental settings, we explored several structures for the network and selected the one that leads to optimal performance in the learning and prediction stages. The parameters are summarised in Table 2.

**Table 2.** Parameters of the LSTM-AE and mlAE.

| Parameter | LSTM-AE | mlAE |
|---|---|---|
| Nodes in input layer | 160 | 160 |
| Neurons in the 1st hidden layer | 120 | 100 |
| Neurons in the 2nd hidden layer | 60 | 50 |
| Neurons in the 3rd hidden layer | 30 | 25 |
| Neurons in the Bottleneck layer | 15 | - |
| Dropout | 20% | - |
| Learning rate | 1e-3 | 1e-3 |
| Batch size | 100 | 50 |
| Number of epochs | 300 | 300 |

Figure 6 illustrates the estimated normal data and anomalies, over time, for the LSTM-AE and the mlAE. The $x$ and $y$ axes, respectively, represent date/time and the value of $RMSE_r$, changes between one (anomaly) and zero (normal). The areas highlighted in pink and light green rectangles, respectively, show the unplanned and the planned failures reported by the company (see Table 3, columns "*Mode*", "*Start time*" and "*End time*"). Since detecting a specific anomaly is not sufficient to conclude a persistent failure, the network generates an alarm when a sequence of anomalies is predicted at least for two

hours. The predicted alarms by the proposed framework are reported under the "*Failure alarm*" column in Table 3.

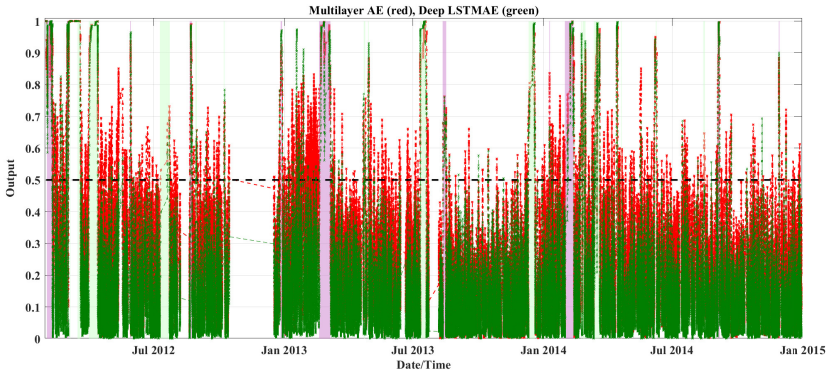


**Fig. 6.** The output of LPF over-time for the LSTM-AE (green) and mlAE (red); data above 0.5 (empirically set) are predicted as anomalies. The pink and light green rectangles indicate the unplanned and planned failures reported by the company.

**Table 3.** Failures reported by the company and start time of failure alarm.

| Nr. | Mode | Start time | End time | **Failure alarm** |
|---|---|---|---|---|
| #1 | planned | 2012-02-02 | 2012-02-09 | 2012-02-01 |
| #2 | planned | 2012-03-01 | 2012-03-02 | 2012-02-29 |
| #3 | unplanned | 2012-03-05 | 2012-03-05 | 2012-03-03 |
| #4 | unplanned | 2012-03-16 | 2012-03-19 | 2012-03-14 |
| #5 | unplanned | 2012-04-01 | 2012-04-13 | 2012-03-31 |
| #6 | planned | 2012-05-29 | 2012-05-29 | 2012-05-29 |
| #7 | unplanned | 2012-07-10 | 2012-07-23 | 2012-07-10 |
| #8 | planned | 2012-08-21 | 2012-08-23 | 2012-08-21 |
| #9 | unplanned | 2012-10-08 | 2012-10-08 | 2012-10-08 |
| #10 | planned | 2012-12-27 | 2012-12-28 | 2012-12-25 |
| #11 | planned | 2013-02-19 | 2013-03-06 | 2013-02-19 |
| #12 | unplanned | 2013-04-23 | 2013-04-24 | 2013-04-22 |
| #13 | unplanned | 2013-04-29 | 2013-04-29 | - |
| #14 | unplanned | 2013-07-11 | 2013-07-19 | 2013-07-11 |
| #15 | planned | 2013-08-12 | 2013-08-16 | 2013-08-12 |
| #16 | unplanned | 2013-12-11 | 2013-12-19 | 2013-12-10 |
| #17 | planned | 2014-01-09 | 2014-01-09 | 2014-01-09 |
| #18 | planned | 2014-01-31 | 2014-02-11 | 2014-01-31 |
| #19 | unplanned | 2014-02-23 | 2014-02-23 | 2014-02-22 |
| #20 | unplanned | 2014-03-13 | 2014-03-20 | 2014-03-12 |
| #21 | unplanned | 2014-04-14 | 2014-04-14 | 2014-04-13 |
| #22 | unplanned | 2014-04-20 | 2014-04-20 | - |
| #23 | unplanned | 2014-06-08 | 2014-06-08 | 2014-06-05 |
| #24 | unplanned | 2014-08-14 | 2014-08-15 | - |
| #25 | planned | 2014-09-03 | 2014-09-05 | 2014-09-03 |
| #26 | planned | 2014-11-28 | 2014-11-28 | - |

From the table, we observe that the predicted date for some of the failures is the same as the date reported for failure ("*Start date*") by the company. Since it is not available at a specific time during the day for the reported failures, it is not possible to compute the exact time (in hours) of the alarms prior to the failures. A detailed comparison between the performance of the two networks is reported in Table 4 where the LSTM-AE obtains a higher Recall, Precision, and F1 Score than those using mlAE.

**Table 4.** Performance comparison of LSTM-AE and mlAE

| | Threshold=0.5 | | Threshold=0.7 | |
|---|---|---|---|---|
| Metric | Multi-layer AE | LSTM AE | Multi-layer AE | LSTM AE |
| $TP$ | 19 | 22 | 20 | 22 |
| $FP$ | 8 | 5 | 4 | 3 |
| $FN$ | 7 | 4 | 6 | 4 |
| $Recall(\%)$ | 73 | 84 | 76 | 84 |
| $Precision(\%)$ | 70 | 81 | 83 | 88 |
| $F1\ Score(\%)$ | 71 | 82 | 84 | 86 |

## 7 Conclusions

We propose a data-driven anomaly detection framework based on deep learning for multivariate time series. We compared two networks, the LSTM-AE and mlAE, which fuse the real-valued data from sensors installed in a bus, compress them and reconstruct to detect anomalies. Raw data are pre-processed to remove noisy data and outliers. Then, statistical parameters of the data are used as features to detect sequences of abnormal operation, since detecting a single instance of abnormal reading is not sufficient to make conclusions about a component failure. Results from analysing the data collected over a period of approximately three years shows that the LSTM-AE has performance superior over that of the mlAE.

Future work includes experiments on rule-based models to explain detected faults and empirically investigate larger datasets, for example from a fleet of vehicles.

## References

1. Chauhan, S., Vig, L.: Anomaly detection in ecg time signals via deep long short-term memory networks. In: 2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA). pp. 1–7. IEEE (2015)
2. de Cheveigné, A., Arzounian, D.: Robust detrending, rereferencing, outlier detection, and inpainting for multichannel data. NeuroImage **172**, 903–912 (2018)
3. Choi, K., Yi, J., Park, C., Yoon, S.: Deep learning for anomaly detection in time-series data: Review, analysis, and guidelines. IEEE Access (2021)

4. Davari, N., Veloso, B., Ribeiro, R.P., Pereira, P.M., Gama, J.: Predictive maintenance based on anomaly detection using deep learning for air production unit in the railway industry. In: 2021 IEEE 8th International Conference on Data Science and Advanced Analytics (DSAA). pp. 1–10. IEEE (2021)
5. Fan, Y., Nowaczyk, S., Rögnvaldsson, T., Antonelo, E.A.: Predicting air compressor failures with echo state networks. In: Third European Conference of the Prognostics and Health Management Society 2016, Bilbao, Spain, 5-8 July, 2016. pp. 568–578. PHM Society (2016)
6. Hines, J., Garvey, D., Seibert, R., , Usynin, A.: Technical review of on-line monitoring techniques for performance assessment. volume 2: Theoretical issues (NUREG/CR-6895, Vol. 2) (2008)
7. Hines, J., Seibert, R.: Technical review of on-line monitoring techniques for performance assessment. volume 1: State-of-the-art (NUREG/CR-6895) (2006)
8. Isermann, R.: Fault-Diagnosis Systems: An Introduction from Fault Detection to Fault Tolerance. Springer-Verlag, Heidelberg (2006)
9. Lei, J., Liu, C., Jiang, D.: Fault diagnosis of wind turbine based on long short-term memory networks. Renewable energy **133**, 422–432 (2019)
10. Lindemann, B., Maschler, B., Sahlab, N., Weyrich, M.: A survey on anomaly detection for technical systems using lstm networks. Computers in Industry **131**, 103498 (2021)
11. Maleki, S., Maleki, S., Jennings, N.R.: Unsupervised anomaly detection with lstm autoencoders using statistical data-filtering. Applied Soft Computing **108**, 107443 (2021)
12. Michau, G., Hu, Y., Palmé, T., Fink, O.: Feature learning for fault detection in high-dimensional condition-monitoring signals (2018)
13. Munir, M., Siddiqui, S.A., Dengel, A., Ahmed, S.: Deepant: A deep learning approach for unsupervised anomaly detection in time series. Ieee Access **7**, 1991–2005 (2018)
14. Nguyen, H., Tran, K.P., Thomassey, S., Hamad, M.: Forecasting and anomaly detection approaches using lstm and lstm autoencoder techniques with the applications in supply chain management. International Journal of Information Management **57**, 102282 (2021)
15. Palai, D.: Vehicle level approach for optimization of on-board diagnostic strategies for fault management. SAE International Journal of Passenger Cars-Electronic and Electrical Systems **6**(2013-01-0957), 261–275 (2013)
16. Ribeiro, R.P., Pereira, P., Gama, J.: Sequential anomalies: a study in the railway industry. Machine Learning **105**(1), 127–153 (2016)
17. Rognvaldsson, T., Nowaczyk, S., Byttner, S., Prytz, R., Svensson, M.: Self-monitoring for maintenance of vehicle fleets. Data Mining and Knowledge Discovery **32**, 344–384 (2018)
18. Venkatasubramanian, V., Rengaswamy, R., Kavuri, S.N., Yin, K.: A review of process fault detection and diagnosis. part I: Quantitative model-based methods. Computers and Chemical Engineering **27**, 293–311 (2003)
19. Zhao, R., Yan, R., Wang, J., Mao, K.: Learning to monitor machine health with convolutional bi-directional lstm networks. Sensors **17**(2), 273 (2017)