Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

Supplementary materials for

Zhenyi XU, Renjun WANG, Yang CAO, Yu KANG, 2023. High-emitter identification for heavy-duty vehicles by temporal optimization LSTM and an adaptive dynamic threshold. *Front Inform Technol Electron Eng*, 24(11):1633-1646. https://doi.org/10.1631/FITEE.2300005

1 Overview of anomaly detection

The problem of identifying high emissions on on-board diagnostic device (OBD) datasets can be transformed into a univariate timing anomaly detection problem. The solutions to this problem can be classified into two types: nonregressive approaches and regression-based approaches.

1.1 Nonregressive approaches

For a stationary time series, the simplest detection method is to manually set high and low thresholds such that when an observation is received outside these bounds, an anomaly is reported.

A more advanced method is to produce a mean and variance for the historic data, define a threshold based on these measures, and report anomalies that fall outside this range (Grubbs, 1969). The box-plot approach is similar, where the data distribution is split into a range of smaller categories and new observations are compared against these ranges (Laurikkala et al., 2000). This may be extended with a larger number of splits, which leads to a histogram approach. These techniques are computationally efficient and require only a small footprint in terms of processor time and memory requirements, but they do not work for most time series because they ignore most of the temporal aspects of the data and treat it as a simple distribution over univariate data. They are therefore unable to detect most contextual and collective anomalies.

Artificial neural networks (ANNs) have also been applied to the problem. Autoencoder neural networks work by taking the values presented in the input layer and passing them into a number of hidden layers with fewer neurons before symmetrically expanding that network toward the output layer. The ability for a trained autoencoder to reconstruct any given input vector gives some insight into how normal that input vector is. A higher reconstruction error suggests that there is some information in the input data that is not expected, given the data used to train that network. Autoencoders were placed on the resource-constrained sensor devices in Luo and Nagarajan (2018). Each device was responsible for collecting sequential data over a period of time and detecting anomalies based on the reconstruction error produced by its shallow autoencoder network. Training was performed in a daily batch method in a central cloud location using the reported input and output vectors generated by each sensor. This relocates the expensive training requirement away from the constrained device and into a more suitable location while reducing the power requirements caused by multiple communications per day.

Recurrent neural networks (RNNs) use feedback loops within the hidden layers in a neural network to allow certain neurons to be affected by outputs from previous time steps, thereby providing some level of memory within the network itself. This allows the network to capture relationships between observations over a period of time. Early RNNs suffered from vanishing gradients, which is difficult in training over large datasets, but with the development of new arrangements of gates, such as long short-term memory (LSTM) and gated recurrent units (GRUs), this problem was mitigated.

An LSTM-based encoder/decoder neural network is employed on a variety of univariate time series where the reconstruction error of the autoencoder is used to identify anomalous sequences within the data (Malhotra et al., 2016). Their method is a semisupervised approach in that the initial network is trained only with normal data. They provide a thresholding mechanism over their computed anomaly score to allow the tuning of the system within a supervised or human-in-the-loop setting based on maximizing the f_β score.

1.2 Regression-based approaches

Another popular approach for identifying outliers is to apply some form of predictive modeling of the time series. A newly received observation is compared against the predicted value and an assessment is made based on the difference between the predicted and actual values.

There are various methods that can be used for the predictive portion of this approach. The autoregressive moving average (ARMA) builds a parametric model of the time series (Geary, 1956). ARMA has seen widespread usage in a number of fields, but this approach has difficulty with nonstationary datasets, in particular, those that display significant seasonality or mean shift. The autoregressive integrated moving average (ARIMA) allows for the management of nonstationarity by adding a number of differencing steps during the processing phase to move the data toward a more stationary distribution (Ventura et al., 2014; Zhu B and Sastry, 2011; Moayedi and Masnadi, 2008; Yaacob et al., 2010; Knorn and Leith, 2008; Bianco et al., 2001). Seasonal ARMA (SARMA) approaches account for differing levels of seasonality within the data by generating multiple models across the different seasonal time lags and apply the same techniques (Kadri et al., 2016).

Another approach to the predictive method is to use ANNs to capture the dynamics of a time series. Early multilayer perceptron (MLP) approaches showed predictive abilities similar to those demonstrated in ARMA derivative models (Zhang and Qi, 2005) for stationary and nonseasonal time series.

ARIMA models are combined with MLPs for predictive analysis, with a simple 2σ thresholding over the error value to identify anomalous observations. The method was demonstrated using electricity consumption data gathered each minute from a university office situation. A very large window size (4 and 8 weeks) was used to generate their models, but full week-ahead predictions were made based on these data. They noted that this method was very sensitive to certain occasional use situations, such as when a printer was in use, which automatically exceeded the 2σ threshold they had selected. Therefore, they introduced some additional rules into the detection engine to compensate for these activities.

With the development of RNNs (Ho et al., 2002; Ghiassi et al., 2005) such as LSTM and GRU, the ability for the neural network approach to better model the variability present in complex univariate systems has been demonstrated in Fu et al. (2016).

An online time-series prediction approach was presented in Guo et al. (2016), whereby the online updating of their LSTM-based neural network was weighted by the loss value from each new data point. When this loss is significant, the algorithm reduces its effect on network updating, thereby minimizing the effect of point anomalies on the predictive capacity of the network, while allowing for change points to be gracefully handled by the network. Although it was not mentioned in this article, there is an opportunity for a pipeline to be developed to allow for these anomalies to be reported to the system operators.

Qin et al. (2017) employed attention-based RNNs within an autoencoder to more accurately predict complex long-term patterns in data.

Malhotra et al. (2015) presented two approaches using stacked layers of recurrent sigmoid units (RSUs) and LSTMs to capture long-term dynamics in various univariate systems. Their networks predict the expected values for a number of time steps ahead, and the resulting error values are used to calculate a probability score that the observation at that later time is within the expected normal range. A threshold value is computed for this probability score and the observations falling below this level are reported as anomalous. They note that for systems with long-term temporal dependencies, the LSTM approach significantly outperforms the RSU approach. A similar deep LSTM network has been applied to ECG signals to identify a variety of different anomalous signals, again using the multiple time-step-ahead probabilistic error measure (Chauhan and Vig, 2015). These approaches both use offline training with a semi-supervised approach.

RNNs are used for regression and two approaches are used for conversion between the raw output and a binary label (Shipmon et al., 2017). Their first method uses thresholding before being passed into an accumulator, which counts up each time an observation is deemed to be anomalous and counts down by a larger factor each time a normal observation is made, thereby detecting collective anomalies due to their longer presentation period. Their second method uses a probabilistic approach to calculate the anomaly likelihood in the most recent observations.

Online time-series anomaly detection using deep RNNs is performed alongside local normalization of the incoming data and incremental retraining of the neural network, to allow the network to adapt to the concept drift across various datasets, showing the applicability of the approach to various domains. Their approach uses the predictive error of the network over a number of time steps to quantify the presence of anomalous observations in a scoring style manner.

While RNNs have shown promise for the prediction of time series, the detection and reporting of anomalous observations based on these predictions is still somewhat a challenge. Xie et al. (2017) presented a method of analyzing the prediction errors using a Gaussian naive Bayes model to process the output of an RNN-based model.

The Greenhouse method computes a vector for each observation using a multistep-ahead predictive RNN (Lee et al., 2018). Their approach uses a three-phase training method. The initial phase fits the RNN to normal data in a typical semisupervised approach. The second phase fits the error vectors generated to distribution, and the final phase calculates Mahalanobis distances between these error vectors to produce a scoring method to identify outliers according to a user-supplied threshold. When presented with a new time series, the algorithm can therefore label each new observation as normal or anomalous based on the post-processed error vector. This approach is currently an offline method and is therefore susceptible to changes in the distributions of the input data.

The RNN model presented by Bontemps et al. (2016) focuses on detecting collective anomalies by defining a minimum period for a collective anomaly, calculating error measurements over time, and identifying an anomaly where the average error is above a given threshold for a period of time.

Zhu L and Laptev (2017) investigated Bayesian neural networks using an LSTM-based autoencoder to perform prediction for a number of steps ahead, followed by an MLP to perform the final prediction steps. This construction provides not only a prediction for later values, but also a level of certainty that when a new observation is made that falls outside a defined predictive interval, the prediction would be flagged as anomalous.

A recent development (Cui et al., 2016a) within the ANN domain is a process described as hierarchical temporal memory (HTM). This process is a bio-inspired model for processing time series based on the behaviors of the neocortex. This method is applied to sequential streamed univariate data and compared against a range of predictive models for time-series modeling (Cui et al., 2016b; Osegi, 2021). The technique is further applied to the anomaly detection problem (Ahmad and Purdy, 2016; Ahmad et al., 2017; Wu et al., 2018; Thill et al., 2017), where it is noteworthy that the noise resistance of the approach and the ability for continual online learning allow the method to adjust to changes in data distribution over time without extensive off-line retraining.

2 Datasets for OBD

2.1 Date stream

The vehicle OBD data streams consist of a combination of several attribute values, including NO_x emission concentration, output torque, and other engine monitoring data. The relevant attribute descriptions are shown in Table S1. The raw OBD data stream is shown in Fig. S1.

Table S1 Description of the properties of the OBD data streams

Attribute	Description
NO_x	Concentration of downstream NO_x (ppm)
Rev	Engine speed, r/min
Torque	Percentage of actual torque output, %
OxyPerc	Percentage of downstream oxygen, %
ExGas	Mass flow rate of exhaust gas, kg/h
Acc	Openness of the gas pedal, %
InsFueInj	Instantaneous engine fuel injection, L
InsFueCons	Instantaneous fuel consumption rate, L/100 km

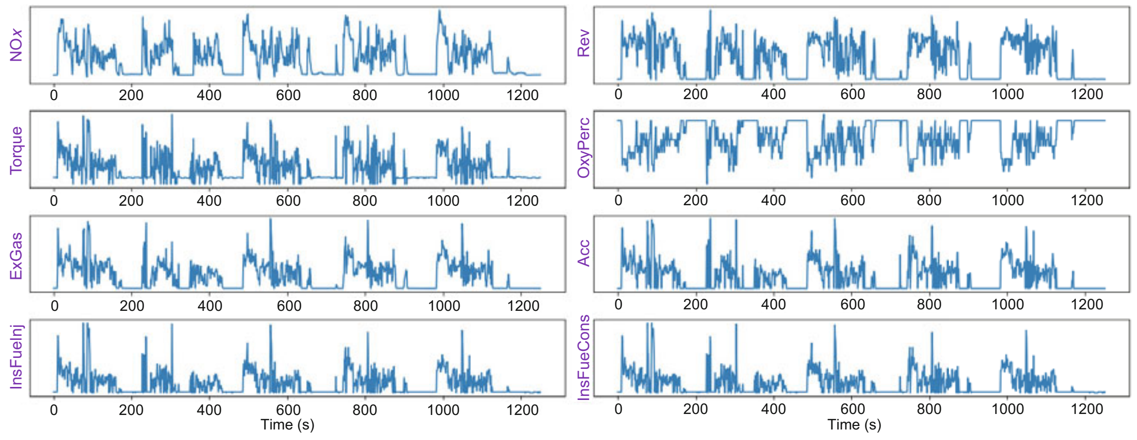

Fig. S1 Raw time-series data stream for OBD

2.2 Detailed experimental data information

In this study, experiments were conducted on OBD data streams from four different motor vehicles, which came from the monitoring system of urban roads in Hefei, China. We used three real vehicle exhaust emission data streams for the evaluation. As shown in Table 1 in the main text, NES denotes the number of the normal emission sequence. Only normal emission sequence data was used for model training and testing. The split ratio of the training set to the test set was 7:3. PES denotes the emission series to be predicted and it was used for model validation. Detailed time information about NES and PES is shown in Table S2.

Table S2 Detailed time information about NES and PES

		OBD1	OBD2	OBD3
Date		2020/6/8	2020/10/25	2020/10/26
NES	Start	20:40:30	07:57:07	10:51:21
	End	21:57:47	09:27:38	11:53:26
PES	Start	22:15:21	14:04:11	15:47:37
	End	23:37:14	16:15:19	16:28:55

NES: number of normal emission sequences; PES: number of emission sequences to be predicted

3 Algorithms and the box plot

3.1 Algorithm

In conjunction with Section 4.2 in the main text, the pseudo code for TSAO-LSTM optimization of time-step attention weights is described in Algorithm S1.

Algorithm S1 TSAO-LSTM optimization of time-step attention weights

Input: (X, y): datasets of time-series operating conditions of vehicle emissions, true concentration labels for NO_x.
Output: M^*: learned TSAO-LSTM model; W^*: attention weight vector of time steps.

```
 1: Population collection: $P \leftarrow \{d_1, d_2, \ldots, d_n\}$
 2: The set of time-step attention weights for $P$: $W \leftarrow \{W_1, W_2, \ldots, W_n\}$
 3: Set of training instances: $F \leftarrow \varnothing$
 4: repeat
 5:     for $i = 1 : n$ do
 6:         $\hat{y} \leftarrow M_i((W_i \cdot X), y)$
 7:         Loss[$i$] $\leftarrow$ the prediction error between $y$ and $\hat{y}$
 8:         Put (Loss[$i$], $M_i$) into $F$
 9:     end for
10:     Select individuals from $F$ according to step (3), then perform crossover and mutation according to step (4)
11:     $P$ $\leftarrow$Restructure a new population following step (5)
12:     Update $W$
13: until up to the maximum number of evolutionary iterations $T$
14: M $\leftarrow$ LSTM model of the individual corresponding to the best fitness value in $F$
15: $W^*$ $\leftarrow$ Time-step attention vector for the individual corresponding to the best fitness in $F$
16: $M^*$ $\leftarrow$ The model combining $M$ with $W^*$
```

Algorithm S2 shows the pseudo code of the similarity metric algorithm between emission prediction error series using DTW (SMEPES).

Algorithm S2 Similarity metric algorithm between emission prediction error series using DTW

Input: $X = [X_0, X_1, \ldots, X_{n-1}] \in \mathbb{R}^n$: concentration prediction error series of NO_x; $Y = [Y_0, Y_1, \ldots, Y_{n-1}] \in \mathbb{R}^n$: concentration prediction error series of NO_x.
Output: S: similarity value between X and Y.

```
 1: Initialize a new matrix $M \in \mathbb{R}^{n \times n}$, which is used to record the similarity values between the different indexes of
    $X$ and $Y$
 2: $M[0, 0] \leftarrow \text{abs}(X_0, Y_0)$
 3: for $i = 1 : n - 1$ do
 4:     $M[i, 0] \leftarrow M[i - 1, 0] + \text{abs}(X_i, Y_0)$
 5: end for
 6: for $j = 1 : n - 1$ do
 7:     $M[0, j] \leftarrow M[0, j - 1] + \text{abs}(X_0, Y_j)$
 8: end for
 9: for $i = 1 : n - 1$ do
10:     for $j = 1 : n - 1$ do
11:         $M[i, j] = \min(M[i - 1, j - 1], M[i, j - 1], M[i - 1, j]) + \text{abs}(X_i, Y_j)$
12:     end for
13: end for
14: return $S \leftarrow M[n - 1, n - 1]$
```

3.2 Box plot

The box plot is a well-known method for data analysis and can be used to screen for outliers (Williamson et al., 1989). First, the values in the sequence are arranged in ascending order. Then, some special values are defined (Fig. S2): the value located at $1/4$ of the sequence is defined as the lower quartile (Q1), the value located at $3/4$ of the sequence is defined as the upper quartile (Q3), and the quartile difference value is defined as $\text{IQR} = \text{Q3} - \text{Q1}$. Finally, the values above $\text{Q3} + 1.5\text{IQR}$ and below $\text{Q1} - 1.5\text{IQR}$ are called outliers. Inspired by the principle of the box plot principle, we consider only values above Q3 as possible outliers in our high-emission timing detection work.

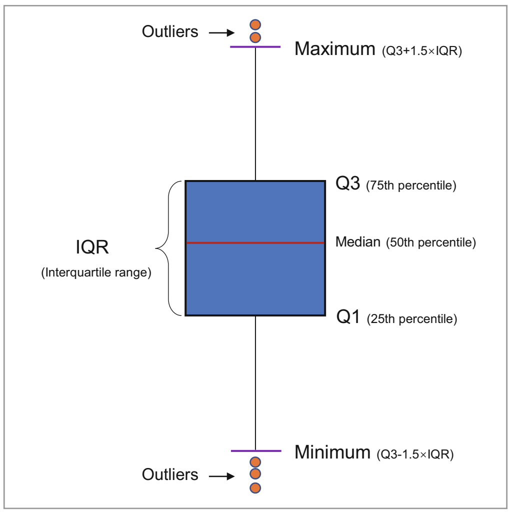

Fig. S2 Principle of the box plot

4 Extended experimental analysis

We conducted additional experiments on our model by dividing the training and test sets in different ratios and performing predictions on the validation set. The experimental results of the three OBD validation sets are shown in Fig. S3. It can be observed that as the ratio of training and test sets increased, the prediction error of the model decreased overall.

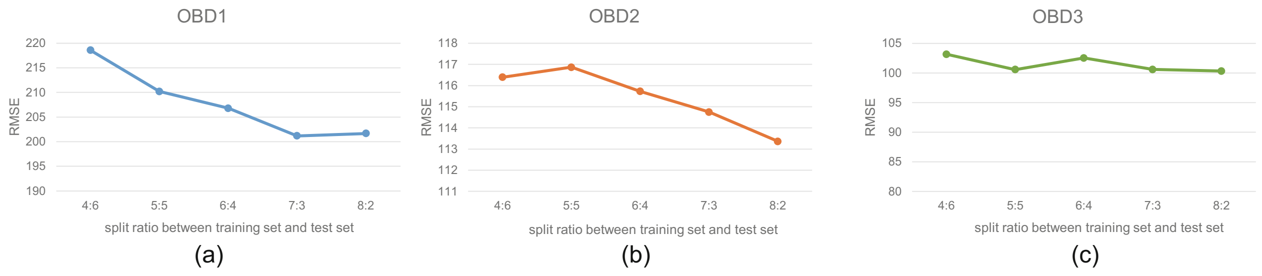

Fig. S3 TSAO-LSTM prediction errors for different split ratios on OBDs: (a) OBD1; (a) OBD2; (a) OBD3

References

Ahmad S, Purdy S, 2016. Real-time anomaly detection for streaming analytics. https://arxiv.org/abs/1607.02480

Ahmad S, Lavin A, Purdy S, et al., 2017. Unsupervised real-time anomaly detection for streaming data. *Neurocomputing*, 262:134-147.

Bianco AM, Garcia BM, Martinez E, et al., 2001. Outlier detection in regression models with ARIMA errors using robust estimates. *J Forecast*, 20(8):565-579.

Bontemps L, Cao VL, McDermott J, et al., 2016. Collective anomaly detection based on long short-term memory recurrent neural networks. Proc 3rd Future Data and Security Engineering, p.141-152.

Chauhan S, Vig L, 2015. Anomaly detection in ECG time signals via deep long short-term memory networks. Proc IEEE Int Conf on Data Science and Advanced Analytics, p.1-7.

Cui Y, Ahmad S, Hawkins J, 2016a. Continuous online sequence learning with an unsupervised neural network model. *Neur Comput*, 28(11):2474-2504.

Cui Y, Surpur C, Ahmad S, et al., 2016b. A comparative study of HTM and other neural network models for online sequence learning with streaming data. Proc Int Joint Conf on Neural Networks, p.1530-1538.

Fu R, Zhang Z, Li L, 2016. Using LSTM and GRU neural network methods for traffic flow prediction. Proc 31st Youth Academic Annual Conf of Chinese Association of Automation, p.324-328.

Geary R, 1956. A study in the analysis of stationary time series. *Econ J*, 66(262):327-330.

Ghiassi M, Saidane H, Zimbra D, 2005. A dynamic artificial neural network model for forecasting time series events. *Int J Forecast*, 21(2):341-362.

Grubbs FE, 1969. Procedures for detecting outlying observations in samples. *Technometrics*, 11(1):1-21.

Guo T, Xu Z, Yao X, et al., 2016. Robust online time series prediction with recurrent neural networks. Proc IEEE Int Conf on Data Science and Advanced Analytics, p.816-825.

Ho SL, Xie M, Goh TN, 2002. A comparative study of neural network and Box-Jenkins ARIMA modeling in time series prediction. *Comput Ind Eng*, 42(2-4):371-375.

Kadri F, Harrou F, Chaabane S, et al., 2016. Seasonal ARMA-based SPC charts for anomaly detection: application to emergency department systems. *Neurocomputing*, 173:2102-2114.

Knorn F, Leith DJ, 2008. Adaptive Kalman filtering for anomaly detection in software appliances. Proc IEEE INFOCOM Workshops, p.1-6.

Laurikkala J, Juhola M, Kentala E, et al., 2000. Informal identification of outliers in medical data. Proc 5th Int Workshop on Intelligent Data Analysis in Medicine and Pharmacology, p.20-24.

Lee TJ, Gottschlich J, Tatbul N, et al., 2018. Greenhouse: a zero-positive machine learning system for time-series anomaly detection. https://arxiv.org/abs/1801.03168

Luo T, Nagarajan SG, 2018. Distributed anomaly detection using autoencoder neural networks in WSN for IoT. Proc IEEE Int Conf on Communications, p.1-6.

Malhotra P, Vig L, Shroff G, et al., 2015. Long short term memory networks for anomaly detection in time series. Proc ESANN, p.89.

Malhotra P, Ramakrishnan A, Anand G, et al., 2016. LSTM-based encoder-decoder for multi-sensor anomaly detection. https://arxiv.org/abs/1607.00148

Moayedi HZ, Masnadi SM, 2008. ARIMA model for network traffic prediction and anomaly detection. Proc Int Symp on Information Technology, p.1-6.

Osegi E, 2021. Using the hierarchical temporal memory spatial pooler for short-term forecasting of electrical load time series. *Appl Comput Inform*, 17(2):264-278.

Qin Y, Song D, Chen H, et al., 2017. A dual-stage attention-based recurrent neural network for time series prediction. https://arxiv.org/abs/1704.02971

Shipmon DT, Gurevitch JM, Piselli PM, et al., 2017. Time series anomaly detection; detection of anomalous drops with limited features and sparse examples in noisy highly periodic data. https://arxiv.org/abs/1708.03665

Thill M, Konen W, Bäck T, 2017. Online anomaly detection on the Webscope S5 dataset: a comparative study. Proc Evolving and Adaptive Intelligent Systems, p.1-8.

Ventura D, Casado-Mansilla D, López-de-Armentia J, et al., 2014. ARIIMA: a real IoT implementation of a machine-learning architecture for reducing energy consumption. Proc 8th Int Conf on Ubiquitous Computing and Ambient Intelligence, Personalisation and User Adapted Services, p.444-451.

Williamson DF, Parker RA, Kendrick JS, 1989. The box plot: a simple visual method to interpret data. *Ann Int Med*, 110(11):916-921.

Wu J, Zeng W, Yan F, 2018. Hierarchical temporal memory method for time-series-based anomaly detection. *Neurocomputing*, 273:535-546.

Xie X, Wu D, Liu S, et al., 2017. IoT data analytics using deep learning. https://arxiv.org/abs/1708.03854

Yaacob AH, Tan IK, Chien SF, et al., 2010. ARIMA based network anomaly detection. Proc 2nd Int Conf on Communication Software and Networks, p.205-209.

Zhang GP, Qi M, 2005. Neural network forecasting for seasonal and trend time series. *Eur J Oper Res*, 160(2):501-514.

Zhu B, Sastry S, 2011. Revisit dynamic arima based anomaly detection. Proc IEEE 3rd Int Conf on Privacy, Security, Risk and Trust and Proc 3rd IEEE Int Conf on Social Computing, p.1263-1268.

Zhu L, Laptev N, 2017. Deep and confident prediction for time series at Uber. Proc IEEE Int Conf on Data Mining Workshops, p.103-110.