Proceedings of the 33rd European Safety and Reliability Conference (ESREL 2023)
Edited by Mário P. Brito, Terje Aven, Piero Baraldi, Marko Čepin and Enrico Zio
 Published by Research Publishing, Singapore.
doi: 10.3850/978-981-18-8071-1_P472-cd

Automated and self-adapting approach to AI-based anomaly detection

Sheng Ding

Institute for Automation Technology and Software System, University of Stuttgart, Germany.
E-mail: sheng.ding@ias.uni-stuttgart.de

Adrian Wolf

Institute for Automation Technology and Software System, University of Stuttgart, Germany.
E-mail: st161569@stud.uni-stuttgart.de

Andrey Morozov

Institute for Automation Technology and Software System, University of Stuttgart, Germany.
E-mail: andrey.morozov@ias.uni-stuttgart.de

Time series anomaly detection (TSAD) is vital across industries, helping identify abnormal patterns to prevent issues, reduce costs, and improve system performance. Nowadays, AI has emerged as a promising solution to enhance TSAD. However, it lacks the self-adapting ability and knowledge of choosing the best-suited model under different contexts. To overcome these challenges, we have integrated various algorithms using a unified data interface and an automated training-testing process. We have incorporated automated hyperparameter optimization and architecture selection. Additionally, we conducted further experiments that demonstrated the advantages of a smart switch mechanism for selecting the most appropriate TSAD method based on statistical features of the data, resulting in improved detection performance. This dynamic switch mechanism has been integrated into our TSAD platform.

Keywords: Fault Detection, Anomaly Detection, Machine Learning, Deep Learning.

1. Introduction

The drive for greater efficiency through automation is a major factor in today's technological advancement, with cyber-physical systems (CPS) playing a crucial role. As these systems become increasingly complex, such as industrial control systems and connected vehicles, manual interception of faults and attacks becomes impractical. As a result, automated anomaly detection is essential to ensure CPS security and safety. Monitoring CPS status is usually achieved through multivariate time series, with each channel representing a separate sensor or communication channel.

An abundance of scientific papers is available presenting new or upgraded methods while claiming to outperform other state-of-the-art methods. The existing research usually takes one or two datasets as the target and then describes the newly invented algorithm. They usually give a list of hyperparameters, which is usually manually decided. Finally, they compare with several older methods over the selected dataset and claim superiority. In short, one dataset along with the proposed algorithm at a time. These "one-at-a-time" workflows are commonly followed by the research community.

However, this workflow has certain drawbacks. First, it is difficult to quickly compare multiple models, and conclusions could potentially be limited to the selected dataset. The lack of generalization and robustness is usually omitted or weakly mentioned. To have an overview of the different efficiency among different algorithms facing different datasets, a better way is to construct everything in an automated manner. Second, even for one certain method, it is also difficult to understand what aspects influence the detection performance to what extent. Therefore it is very hard to keep track of the actual progress in this field.

Contributions: First, for the purpose of overcoming the previously mentioned challenges, we integrate different algorithms together with a unified data interface, automated configuration, and

training-testing process. Second, we have done additional experiments to prove a better mechanism than relying on a single TSAD method. A smart switch mechanism capable of selecting a method based on statistical features of the data can lead to better detection performance. Third, our tool and corresponding experiments are open-source for the research community.[a]

As illustrated in Fig.1, we build an anomaly detection platform that consists of different steps and corresponding components. All of these have different kinds of possibilities. As a result, users can choose different combinations of models, thresholding functions, and evaluation metrics.

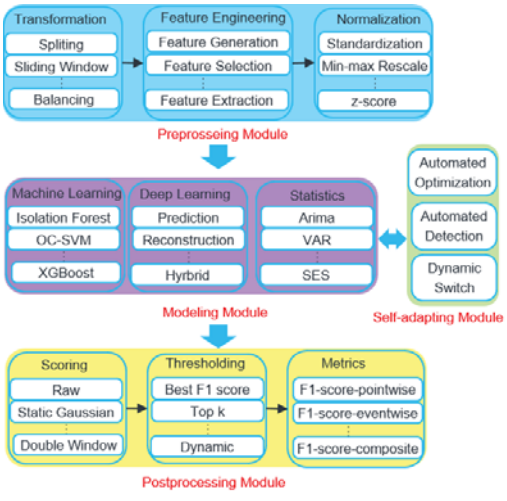

Fig. 1.: Automated anomaly detection with modularized design, Preproccessing module (blue) is described in Sec. 2, Modeling module (purple) is described in Sec. 3, Postprocessing module (yellow) is described in Sec. 4, Self-adapting module (green) is described in Sec. 5

2. Preprocessing module

2.1. *Data set*

The input of the platform is defined by a data interface. It enables the import of any time series dataset stored in CSV files, where values are separated by commas and the dot is used as the decimal separator. Several time series datasets from different CPSs were collected. The origins and attributes of these five datasets are briefly presented in the following.

NASA Soil Moisture Active Passive satellite (SMAP) This expert-labelled real-world dataset was collected from "Incident Surprise, Anomaly" (ISA) reports from NASA's Soil Moisture Active Passive satellite Hundman et al. (2018). It consists of 55 different physical entities of equal type and dimensionality. Each entity contains 25 channels with only one channel being a sensor value while all other channels represent commands received by that entity.

NASA Mars Science Laboratory (MSL) Similar to the SMAP dataset, this real-world dataset was expert-labelled using ISA reports for NASA's Mars Science Laboratory, the Curiosity rover Hundman et al. (2018). It also consists of multiple entities (27) with multiple channels each (55). As for the SMAP dataset, only the single sensor channel is considered in this work containing the same types of anomalies.

Unmanned Aerial Vehicle (UAV) This univariate dataset was synthetically generated using a Simulink model. Of the two available entities, the accelerometer and the gyroscope, only the gyroscope dataset was used for evaluations in this work. The dataset was injected with three types of anomalies: Offset, noise and stuck-at faults.

Automated Vehicle System (AVS) Similar to the UAV dataset, it was generated with fault injection experiments using Simulink models. In contrast to the UAV dataset, the anomalies in each time series are separated by a fixed interval.

Server Machine Dataset (SMD) The Server Machine Dataset Su et al. (2019) is a multivariate dataset. It was collected over five weeks and contains 28 different entities of three groups. Each of the entities consists of 38 individual channels.

2.2. *Data spliting*

Figure 2 shows the methods of splitting the dataset. The fault-free data gets split into a train set Z_{FF} and a training validation set V_{FF}. The anomalous test set gets split into 90% test set

[a] www.github.com/mbsa-tud/tsad_platform

T_A and 10% anomalous validation set V_A. Since an anomalous validation set might however not always be available and many datasets just contain a single anomaly in their test data, the splitting of the test set into T_A and V_A is only done for datasets with multiple files for testing, such that at least one of the files is used for V_A. Otherwise, the entire test set is used as V_A.

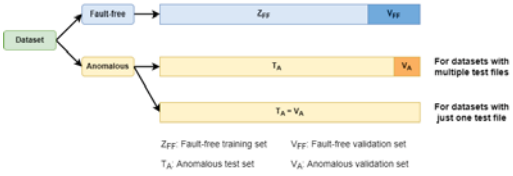

Fig. 2.: Data preparation for the TSAD platform.

After splitting a dataset into the aforementioned subsets, the data is further prepared for the TSAD methods by splitting it into overlapping subsequences/windows of equal length w separated by a fixed step-size.

2.3. *Feature extraction*

In TSAD, feature generation, selection, and extraction are essential components of the overall process. Feature generation involves creating new features based on the available data. A variety of techniques can be employed, including statistical measures such as mean and skewness. These features are used to capture the distribution and central tendency of the data. Feature selection is performed to identify the most relevant features. This is typically done to reduce the dimensionality of the feature space and improve computational efficiency. Feature extraction is used to transform the selected features into a more compact and representative feature space.

2.4. *Normalization*

Normalization is a critical step in preparing time series data for anomaly detection. The process involves scaling the data to ensure that all features have the same weight during analysis. There are various techniques for normalizing time series data, including min-max normalization, Z-score normalization, and robust scaling. Different normalization methods may perform better under different conditions, and the selection of the appropriate method can significantly impact the accuracy of the anomaly detection model.

3. Modeling module

3.1. *Integrating different TSAD methods*

After literature research on different TSAD methods, we find out that there are different ways of categorizing them. First, they can be grouped into statistical, machine learning, and deep learning methods. Second, they can be grouped into semi-supervised, unsupervised, and supervised methods. Third, depending on the principle of how they detect anomalies, they can be grouped into methods such as predictive, reconstructive, distance-based, and isolation tree-based methods. As users are usually more familiar with the first categorization, we provide the user interface with the first categorization, as shown in Fig.3. However, for the purpose of integrating different TSAD methods, the second one is more essential to differentiate their training and testing procedure.

3.1.1. *Semi-supervised methods*

Semi-supervised methods are trained only on normal data without anomalies. Representative methods in this regard are Predictive methods and Reconstructive methods, most of these methods exploit deep learning neural networks. **Predictive methods** compute the anomaly scores θ by predicting the k next values following a given subsequence. Afterwards the predicted points are compared to the observed points. Such methods include convolutional neural networks (CNN) like the DeepAnT Munir et al. (2018) architecture, or long-short term memory (LSTM) networks Hundman et al. (2018). In Braei and Wagner (2020), a multilayer perceptron (MLP) and a residual network (ResNet) are also used to make predictions. **Reconstruction methods** on the other hand encode the subsequences into a latent space representation and try to reconstruct it from there. The difference between the reconstructed and observed points in the subsequence results in the anomaly score θ. Examples are autoencoders like fully connected layers used in Sakurada and Yairi

(2014) or the temporal convolutional autoencoder (TCN-AE) Park et al. (2022).

3.1.2. *Unsupervised methods*

Unsupervised methods don't require prior training on anomalous data. Instead, they operate under the assumption that anomalous points or subsequences can be distinctly separated from the rest of the data. **Distance methods**, as an example, work under the assumption of anomalous points or subsequences being further away from normal ones. Examples are the local outlier factor (LOF) Breunig et al. (2000), the local distance-based outlier factor (LDOF) Zhang et al. (2009), angle based outlier detection (ABOD) Kriegel et al. (2008).

3.1.3. *Supervised methods*

Supervised approaches use labeled data during training containing normal points and anomalies and learn to classify points as normal or anomalous. We have integrated several DL-based classification methods. However, due to their reliance on the availability of massive labeled data, these methods may not be truly effective.

3.2. *User Interface*

3.2.1. *Training and detection*

The main training panel of the TSAD platform can be seen in Fig. 3. DNN, classic ML and statistical models can be added to separate lists. This can be done either by adding a default configuration, which is stored in a JSON file, or by manually adjusting the different hyperparameters within the corresponding configuration panels. Users can add as many models to these lists as needed and even include the same model multiple times with different configurations. At the bottom of these lists, the training and optimization buttons are located. It's possible to train either all models or a selection of models from the list consecutively. For DNNs, the training can also be done in parallel.

3.2.2. *Model recommendation*

Once the models have been trained, they appear in the list of trained models on the left-hand side of the detection panel, as shown in Fig.4. Users can either run the detection process on a selection of the trained models or for all models consecutively. Afterwards, the list of trained models gets sorted to show the best-performing model on top. This ranking is done by comparing the model's F_1 scores for the selected static threshold. Additionally, another metric can be chosen to rank the models by. Next to the model ranking, the results for the selected threshold are depicted and the parameters for the dynamic threshold can be reconfigured.

Fig. 3.: Training panel of the platform.

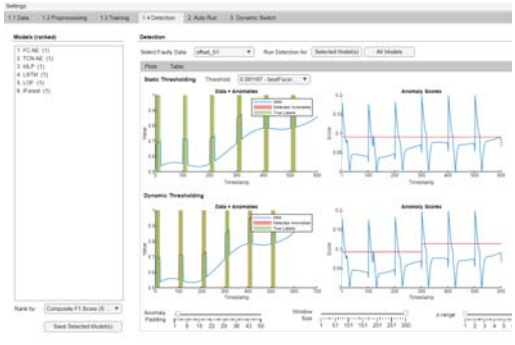

Fig. 4.: Detection panel of the platform.

4. Postprocessing module

After getting the output of the modeling module, the platform computes anomaly scores using different anomaly functions, including static and dynamic gaussian scoring methods. The goal is to reduce noise which is a potential cause of false alarms. Then the anomaly score is passed to the thresholding function to get the binary detection.

4.1. *Thresholds*

Various static thresholds and a non-parametric dynamic threshold are available to transform anomaly scores into binary labels.

4.1.1. *Best F score threshold*

Best F score thresholds are calculated on anomalous validation set. If this set is not available, they are calculated during testing after running the detection.

4.1.2. *Top-k threshold*

The newly added static top-k threshold also sets a threshold value with the support of a labeled anomalous validation set V_A, similar to the aforementioned best F_1 score thresholds. The difference however is, the top-k threshold gets set as such, that exactly k points of V_A are labeled as anomalous, k being the number of anomalous observations in V_A.

4.1.3. *Dynamic threshold*

The non-parametric dynamic thresholding method was adapted from Hundman et al. (2018) and uses a windowing approach. The prediction/reconstruction errors are smoothed using the exponentially weighted moving average (EWMA) before assigning each window a separate threshold from a predefined range.

4.2. *Evaluation metrics*

Various evaluation metrics can be used to judge the detection performance of a TSAD method. Table 1 shows all available evaluation metrics that can be calculated by the platform. In addition to the point-wise (weighted) and event-wise (unweighted) metrics, six new metrics were introduced: Point-adjusted metrics, as proposed by Xu et al. (2018), and composite F_β scores, which were initially introduced by Garg et al. Garg et al. (2021). The value of all metrics can reach from 0 to 1, 0 being the worst possible score and 1 the best.

Metric	Point-wise	Event-wise	Point-adjusted	Composite
Precision	Pr	Pr_e	Pr_p	-
Recall	Rec	Rec_e	Rec_p	-
F_1 score	F_1	F_{e1}	F_{p1}	F_{c1}
$F_{0.5}$ score	$F_{0.5}$	$F_{e0.5}$	$F_{p0.5}$	$F_{c0.5}$

Table 1.: Available evaluation metrics.

4.2.1. *Point-wise metrics*

Point-wise metrics compare a model's detected labels to the ground truth labels for each individual observation in the time series. This is done by initially calculating the number of true positive (TP), false positive (FP) and false negative (FN) points. Afterwards, common point-wise metrics, including precision (Pr), recall (Rec) and the F_β score, are computed:

$$Pr = \frac{TP}{TP + FP} \tag{1}$$

$$Rec = \frac{TP}{TP + FN} \tag{2}$$

$$F_\beta = (1 + \beta^2)\frac{Pr \cdot Rec}{(\beta^2 \cdot Pr) + Rec} \tag{3}$$

Only the F_1 score, which is the harmonic mean of precision and recall, and the $F_{0.5}$ score, which assigns a higher weight to the precision, are computed by the platform.

4.2.2. *Event-wise metrics*

The main difference between event-wise and point-wise metrics is, event-wise metrics just consider true and falsely detected events. An event is a continuous sequence of anomalous points. If a detected sequence overlaps with a true anomalous sequence in any way, that sequence is counted as a true positive TP_e. If it doesn't overlap with any true sequence, it is counted as a false positive FP_e, and if for a true sequence no overlapping detected sequence is found, a false negative FN_e is recorded. Using these values, the precision Pr_e,

recall Rec_e, F_{e1} score and $F_{e0.5}$ score are computed similar to the point-wise metrics.

4.2.3. *Point-adjusted metrics*

For the point-adjusted metrics, if a single true positive is found within an anomalous segment, all detected points in this segment are set to 1. Afterwards, the precision Pr_p, recall Rec_p, F_{p1} score and $F_{p0.5}$ score are calculated in the same way as for the point-wise scores. These metrics can therefore be interpreted as version of the point-wise metrics which assign a single true positive in an anomalous segment a way higher value.

4.2.4. *Composite metrics*

The new composite F_β score is similar to the common F_β score with one major difference. It combines the point-wise precision Pr and event-wise recall Rec_e as follows:

$$F_{c\beta} = (1+\beta^2)\frac{Pr \cdot Rec_e}{(\beta^2 \cdot Pr) + Rec_e} \qquad (4)$$

This platform offers both the F_{c1} score, which is the harmonic mean of Pr and Rec_e, and the $F_{c0.5}$ score, which assigns Pr a higher weight than Rec_e.

5. Self adapting module

5.1. *Automated recommendation*

In current research on TSAD, typically one or two datasets are selected as the focus, a new algorithm is introduced with manual configuration, and then the algorithm is compared to several older methods with the claim of superiority. These conclusions may be restricted to the specific dataset and hyper-parameter configuration used, with limited emphasis placed on generalization and robustness. To obtain a comprehensive overview of algorithmic efficiency across diverse datasets (context), a more effective approach is to construct a fully automated system that includes training and hyperparameter optimization, as well as automated detection and evaluation. With such a system, state-of-the-art methods can be automatically compared over user-selected datasets to recommend the optimal solution.

5.1.1. *Automated hyperparameter optimization*

It's important to optimize the entire training and detection process by finding a set of hyperparameters that results in the best score. Common approaches like grid search or random search choose several combinations of hyperparameters from a predefined grid or range of possible values. These methods might however take a very long time to find optimal hyperparameters as they evaluate many different combinations in a brute-force way. The default optimization algorithm for this platform is bayesian optimization. It aims to solve the following problem: $max(f(x)), x \in A$, $f(x)$ is referred to as the objective function, for which the maximum value should be found. In the TSAD platform, the objective function encapsulates the entire process of training and testing a model. It receives a combination of hyperparameters as input and outputs the computed metric for the test set. The algorithm operates as follows:

(1) $f(x)$ is evaluated for a random selection of values x_0.
(2) A Bayesian statistical model providing a posterior probability distribution, which describes potential values for $f(x)$ at point x, is updated.
(3) An acquisition function selects a new point x to be evaluated next. It chooses the next point for which it expects to improve $f(x)$ the most when compared to the current best point x_{best}.
(4) Step 1 and 2 are repeated N times and the best point x_{best} is returned.

With the help of the optimization popup window provided by the platform, it is possible to define the range of hyperparameters, choose the metric to optimize the model for, and set the number of iterations N for the bayesian optimization.

5.1.2. *Automated detection and evaluation*

The platform's auto-run panel offers configuration options to train and test a selection of models on an entire dataset. The workflow for configuring and running the automated evaluations can be summarized as such:

(1) A dataset folder has to be selected.

(2) A preprocessing method has to be chosen from the drop-down menu.
(3) A selection of models has to be made. They can either be configured on the training panel or by creating and importing a json file.
(4) The different thresholds to be used need to be selected in the threshold selection panel.
(5) The evaluation gets started by pressing the "Run Evaluation" button, a folder gets created within the current folder containing all generated results stored in CSV files.

5.2. *Dynamic switch*

A few surveys about TSAD methods Garg et al. (2021); Audibert et al. (2022) have made comparisons and discussions. They state that no model seems to fit all contexts, and this motivates the implementation of the dynamic mechanism. The dynamic switch is a model selection mechanism, which automatically chooses the best-suited model for the current statistical features of the time series based on previously obtained knowledge. This leads to a better detection performance overall, since the restriction to a single model, which could be good for some contexts but worse for others, exists no longer.

5.2.1. *Workflow for dynamic switching of best-suited model*

In Figure. 5, the workflow for training and testing the model selection mechanism of the dynamic switch can be seen. The mechanism's two major elements are the feature extraction process and the classifier. The dynamic switch can be configured, trained, and tested on the platform's dynamic switch panel. The five different steps that make up the workflow of the dynamic switch (see Figure 5) are as such:

(1) A dataset containing many different anomalous time series is chosen and split randomly into a training set and a test set.
(2) A set of TSAD models is selected. All models get tested on every time series of the training set. Each time series gets labeled by finding the corresponding best-performing model.
(3) For each labeled time series of the train set, multiple statistical time series features are extracted. Tuples of feature vectors and their corresponding label representing the best model for each time series are generated.
(4) The prepared data is used to train a classifier. The network learns to output the correct labels for a given set of features.
(5) In the testing phase, the trained classifier recommends a model according to the newly extracted features. This model is then used to detect anomalies in the corresponding time series.

6. Conclusion

TSAD research using only selected datasets and manual configuration can result in limited generalization and robustness. We introduce a concept for an automated system that includes training, hyperparameter optimization, and evaluation across diverse datasets. This allows for automatic comparison of state-of-the-art methods and recommends the optimal solution.

References

Audibert, J., P. Michiardi, F. Guyard, S. Marti, and M. A. Zuluaga (2022). Do deep neural networks contribute to multivariate time series anomaly detection? *arXiv preprint arXiv:2204.01637*.

Braei, M. and S. Wagner (2020). Anomaly detection in univariate time-series: A survey on the state-of-the-art. *arXiv preprint arXiv:2004.00433*.

Breunig, M. M., H.-P. Kriegel, R. T. Ng, and J. Sander (2000). Lof: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pp. 93–104.

Garg, A., W. Zhang, J. Samaran, R. Savitha, and C.-S. Foo (2021). An evaluation of anomaly detection and diagnosis in multivariate time series. *IEEE Transactions on Neural Networks and Learning Systems 33*(6), 2508–2517.

Hundman, K., V. Constantinou, C. Laporte, I. Colwell, and T. Soderstrom (2018). Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. In *Proceedings of the 24th ACM SIGKDD international con-*

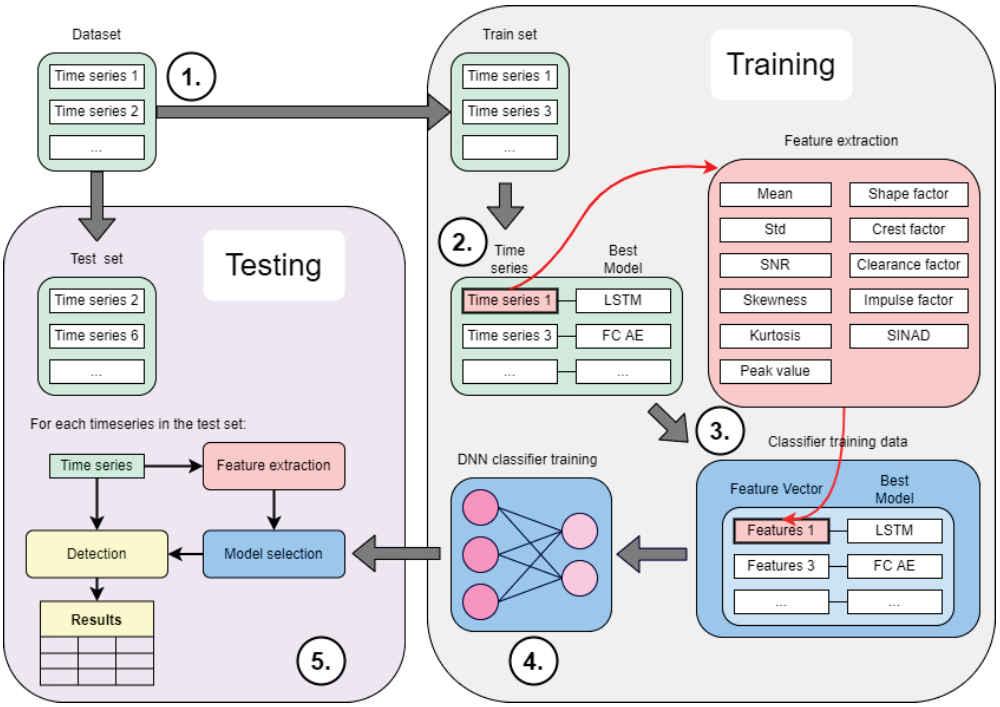

Fig. 5.: Workflow for the model selection mechanism of the dynamic switch.

ference on knowledge discovery & data mining, pp. 387–395.

Kriegel, H.-P., M. Schubert, and A. Zimek (2008). Angle-based outlier detection in high-dimensional data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 444–452.

Munir, M., S. A. Siddiqui, A. Dengel, and S. Ahmed (2018). Deepant: A deep learning approach for unsupervised anomaly detection in time series. *Ieee Access 7*, 1991–2005.

Park, J., Y. Park, and C.-I. Kim (2022). Tcae: Temporal convolutional autoencoders for time series anomaly detection. In *2022 Thirteenth International Conference on Ubiquitous and Future Networks (ICUFN)*, pp. 421–426.

Sakurada, M. and T. Yairi (2014). Anomaly detection using autoencoders with nonlinear dimensionality reduction. In *Proceedings of the MLSDA 2014 2nd workshop on machine learning for sensory data analysis*, pp. 4–11.

Su, Y., Y. Zhao, C. Niu, R. Liu, W. Sun, and D. Pei (2019). Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 2828–2837.

Xu, H., W. Chen, N. Zhao, Z. Li, J. Bu, Z. Li, Y. Liu, Y. Zhao, D. Pei, Y. Feng, et al. (2018). Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. In *Proceedings of the 2018 world wide web conference*, pp. 187–196.

Zhang, K., M. Hutter, and H. Jin (2009). A new local distance-based outlier detection approach for scattered real-world data. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 813–822. Springer.