DOI: 10.15439/2022R04 ISSN 2300-5963 ACSIS, Vol. 33

# udCATS: A Comprehensive Unsupervised Deep Learning Framework for Detecting Collective Anomalies in Time Series

1st Truong Son Pham
*Faculty of Information Technology*
*Le Quy Don Technical University*
Hanoi, Vietnam
0000-0001-6320-8501

2nd Viet Hung Nguyen
*Faculty of Information Technology*
*Le Quy Don Technical University*
Hanoi, Vietnam
0000-0002-9818-4455

3rd Anh Thang Le
*Faculty of Information Technology*
*Le Quy Don Technical University*
Hanoi, Vietnam
0000-0003-2019-4781

4th Van Duong Bui
*Faculty of Information Technology*
*Le Quy Don Technical University*
Hanoi, Vietnam
0000-0002-0905-5214

***Abstract*—Anomaly detection has recently gained enormous attention from the research community. It is widely applied in many industrial areas, such as information security, financing, banking, and insurance. The data in these fields can mainly be represented as time series data, the corollary being that time series anomaly detection plays an essential role in these applications. Therefore, many authors have tried to solve the problem of collective anomaly detection in time series. They have proposed several approaches, from classical methods such as Isolation Forests to modern deep learning networks such as Autoencoders. However, a comprehensive framework for handling this problem is still lacking. In this work, firstly, we propose using an Attention-based Bidirectional LSTM Autoencoder (Att-BiLSTM-AE) as an anomaly detection model. Furthermore, in the essential part of this paper, we developed a comprehensive unsupervised deep learning framework, udCATS, to solve the problem of detecting collective anomalies in time series. Our experiments show that the Att-BiLSTM-AE outperforms other detection models, and using it within the udCATS framework increases the detection accuracy.**

***Index Terms*—collective anomaly, time series, unsupervised, deep learning**

## I. Introduction

Anomaly detection plays an essential role in many industrial areas, for example, financing, banking, information security, and insurance. Many data in these domains can be represented as time series. Because of that, anomaly detection in time series data has recently gained massive attention from the research community.

A time series can be univariate or multivariate, discrete or continuous. In this work, we focus only on discrete univariate time series. Therefore, the term "time series" used in the rest of this article refers to a discrete univariate time series. **Time series** by its definition, is a set of data collected at successive, discrete timestamps and can be written as $\{X_t,\ t \in Z\}$ [1]. The term anomaly of a time series can be considered an outlier. From the traditional point of view, an **outlier/anomaly** is an observation that varies "extensively from the other one as to produce suspicions that it was generated by a different mechanism [2]."

An anomaly in time series can deliver important information. For example, it could be some unwanted data points that were produced or collected incorrectly. In this case, anomaly detection is essential for data cleaning, which is crucial for developing proper machine learning models. In addition, the anomaly can also represent the events of interest, such as machine breakdowns, cyber-attacks, and insurance frauds, which are the main applications of anomaly detection in time series.

The anomalies in time series can be divided into three main categories: point, collective, and contextual anomaly [3]. A time series data point is an anomaly when it behaves out of the ordinary compared to most other points. The term collective anomaly refers to consecutive data points with unusual behavior. It is crucial to mention that each point of an abnormal sub-sequence is not necessary an outlier. Contextually anomaly is used when some time series points are typical in a specific context but anomalous in another context [3].

We focus here on collective anomaly detection because detecting the collective outliers is much more challenging than detecting the unusual points. As mentioned above, a single data point in a sub-sequence may not be an outlier; however, they will build up an abnormal sub-sequence when considering them in consecutive order. That makes the research problem much more challenging. Besides that, the problem of point anomaly detection is already well-researched [4]. In contrast, the detection accuracy can still be improved in the problem of collective anomaly detection by proposing or applying contemporary deep learning networks. In our work, firstly, we propose using an attention-based bidirectional LSTM Autoencoder (Att-BiLSTM-AE) as an anomaly detection model. Furthermore, in the essential part of this paper, we developed a comprehensive unsupervised deep learning framework called udCATS to solve the problem of detecting collective anomalies in time series. Our experiments show that the Att-BiLSTM-AE outperforms other detection models while using it within the udCATS framework increases the detection accuracy.

The rest of this paper is organized as follows. First, section II concerns some selected unsupervised learning approaches to detect collective anomalies. Next, the udCATS framework, which includes four primary processes, is described in Section III. Finally, section IV details our experiments and discusses their results before we clarify in Section V how we would like to improve the framework continually.

## II. Related Work

Many methods and approaches have been proposed to detect collective anomalies in time series. They can be grouped into two categories: supervised and unsupervised detection methods. In comparison, the approaches can be divided into three groups: statistical, classical machine learning, and deep learning models [3].

Supervised methods typically produce increased detection precision; however, they are pretty unuseful because they require labeled data sets, which are usually unrealistic. The labeling process is nowadays one of the most costly steps in a Machine Learning Pipeline. On the other hand, unsupervised methods are much more practical and valuable. However, receiving a high accuracy with unsupervised learning models is very demanding. Deep learning models have demonstrated their robustness and accuracy in an unsupervised manner compared to statistical and classical machine learning models [5], [6]. In this section, unsupervised approaches applied for collective anomaly detection problems and time series are discussed briefly [6]–[11].

One of the most straightforward ideas to detect the anomalies in an unsupervised manner is applying clustering algorithms such as K-Means Clustering [8] or Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [9]. The detailed descriptions of these clustering algorithms are provided by [12], [13] and [14].

C. Mete, F. Dadaşer-Çelik, and A. Dokuz [9] applied DBSCAN to detect anomalies in a dataset that contains the daily average temperature over 33 years. The author segmented the time series into monthly sequences, normalized them by their mean and variance, and then clustered them with DBCAN. The results show that DBSCAN can detect collective anomalies even if there is no significance between them and the usual data points. Keogh and Lin [8], nevertheless, have indicated that using clustering algorithms for collective anomalies detection is senseless. They showed that the cluster centers discovered for several runs of the K-means algorithm on the same dataset are not remarkably contrasting to the one of a random walk process. Some authors tried to analyze and overcome this problem. However, it remains unsolved [15].

L. Bontemps, V.L. Cao, J. McDermott and N.A. Le-Khac [7] proposed a LSTM-based collective anomaly detection model. Firstly, the time series is modeled with an LSTM RNN [16]. The predictive model is then adapted to propose a circular array containing prediction errors from several recent time steps. Finally, a predetermined threshold is applied to indicate a collective anomaly. To evaluate the model, the authors converted the KDD 1999 dataset [17] into a time series version. The results showed that without any false alarm, the model could detect 86% of the collective anomalies. If the threshold is set to capture all the anomalies, the number of false alarms is increased to 63.

Besides LSTM Network, some other deep learning models are also proposed for detecting collective anomalies in time series, such as Convolutional Neural Networks (CNN) [6], Gated recurrent unit (GRU) [10], and Autoencoder [11]. The results show that, in general, deep learning models perform very well for collective anomaly detection problems in time series data.

We can make some important conclusions based on the knowledge gained from a comprehensive literature review, especially from the selected publications discussed above:

- There is still no comprehensive framework for detecting time series collective anomalies. The task of detecting collective anomalies is not trivial as putting the time series into a detection model to get the results. It requires several steps, for example, splitting the time series into sub-sequences, reducing the data dimension, scaling the features, etc.
- Clustering-based approaches are not suitable for this kind of problem.
- Deep learning models produce highly accurate results when solving the problem of collective anomaly detection.

For these reasons, we propose a comprehensive framework, called udCATS, for detecting collective anomalies in time series in an unsupervised manner. The framework uses an Attention-based Bidirectional Long Short-Term Memory Autoencoder as the anomaly detection engine. All the components of the udCATS framework are essential for solving the problem.

## III. udCATS Framework

This section explains the udCATS framework in detail. It first clarifies the architecture and then each component of the framework.

### A. *Framework Architecture*

The framework contains four components: time series segmentation, representation, scaling, and anomaly detector engine. The time series is first segmented into sub-sequences, later transformed to reduce the high dimensionality. These processes are called segmentation and representation. The output of the representation process is then used as the input for the data scaling process. In the end, an Attention Bidirectional

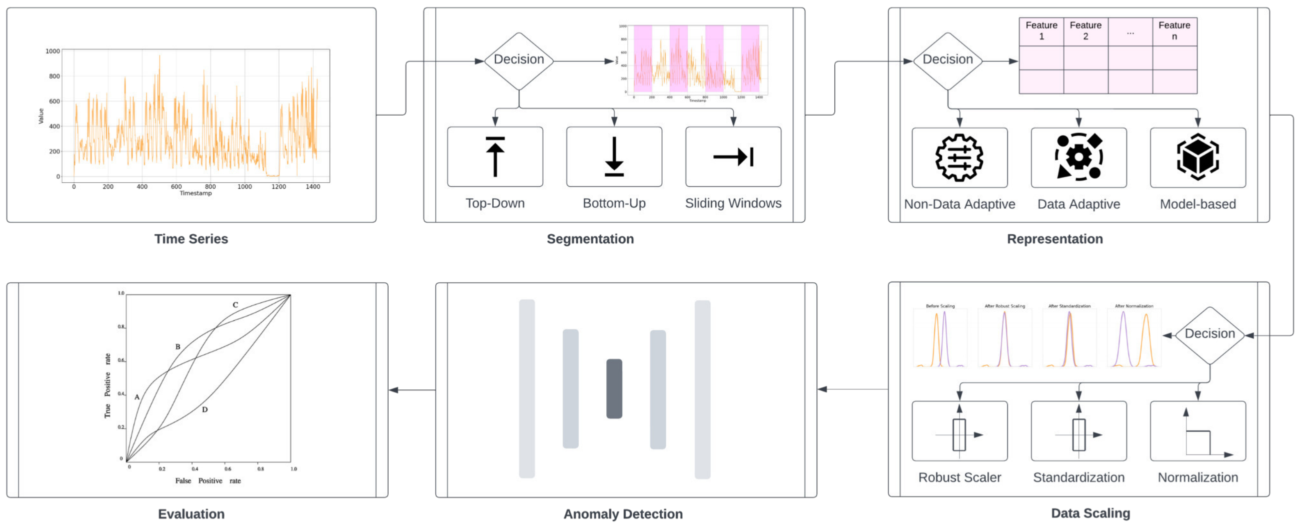

Fig. 1. Architecture of the udCATS Framework

Long Short-Term Memory Autoencoder is used to detect abnormal samples. For example, suppose a sample is classified as an anomaly. In that case, it can be used to identify the original sub-sequence to determine the collective abnormalities. Figure 1 illustrates the architecture of the udCATS framework.

Each of the components mentioned above is a selection process, which means different methods can be selected based on the nature of the input time series. For time series segmentation, top-down, button-up, or sliding windows can be selected, while non-data adaptive, data-adaptive, and model-based approaches are the most prominent time series representation approaches. Data-dictated representation can also be discovered in the literature. However, it is not widely used for this task. We experimentally recommend an Attention Bidirectional Long Short Term Memory Autoencoder as an anomaly detection engine. Although it is not mandatory, another deep learning network can also be used for this part. It depends, as explained, on the nature of the input data. Last, udCATS establishes standardization, normalization, and robust scaling for the data scaling process.

The remainder of this section expresses each element of the framework in detail.

## *B. Time Series Segmentation*

Time series segmentation is a method of time-series analysis in which an input time series is divided into a sequence of discrete segments, called sub-sequence, to reveal the underlying properties of its source [18]. An optimal segmentation algorithm is defined as the one with minimal approximation error, calculated based on the difference between the segmented sub-sequences and the original time series. Figure 2 visualizes the segmentation process of the proposed udCATS framework. This is inspired by the work of M. Lovric, M. Milanovic, and M. Stamenkovi [18].

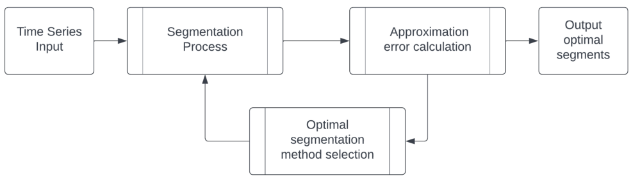
Time Series Input
Segmentation Process
Approximation error calculation
Output optimal segments
Optimal segmentation method selection

Fig. 2. The time series segmentation process

The following paragraphs describe the most well-known segmentation algorithms: sliding windows, top-down, and bottom-up [18].

**Sliding Windows**, also called ”brute-force” or ”one-pass” algorithm [18], it is one of the most widely involved time series segmentation algorithms. It starts with appointing the first data point as the anchor. Afterward, the window size is initially determined, and based on this size, the approximation error for the potential segment is calculated. Next, the window size is increased until the approximation error exceeds a predetermined threshold. Finally, a segment is created with the possible largest window size. This process is repeated until the sliding windows are across the entire time series. The new anchor is updated as the next data point right behind the created segment.

The **Top-Down** algorithm considers the original time series as one major segment. It starts with finding the breaking point, which divides the time series into two parts with the maximal difference between them. The approximation error is then calculated for both segments and compared with the predetermined threshold. These steps are repeated for all of the segments until the approximation error exceeds the threshold [18].

The **Bottom-Up** algorithm is the opposite of the top-down algorithm described above. It starts with segmenting the time series of length $n$ into $n-1$ segments. Then, a segment is decided to merge with the one on the left or the right based on increasing the approximation error. Finally, it takes the one with a minor error increase. The merging process is repeated until the approximation error of a segment exceeds a predetermined threshold [18].

### C. Time Series Representation

Unsupervised detection methods often do not directly use the original time series data points as the input. Instead, representations of the time series will be used. The representation is helpful for dimension reduction and similarity measurement and often helps produce better results [19].

There are four main approaches to time series representation: non-data adaptive, data-adaptive, model-based, and data-dictated representation [20]. The parameters can be fine-tuned with the first three approaches to find the best time series compression for the particular application. However, the time series dictates the compression itself with the last one. For this reason, only non-data adaptive, data-adaptive, model-based approaches are used for the selection process of the time series representation process.

In **non-data adaptive** algorithms, the represented parameters remain the same for all time series, independent of their nature. Some of the most widely used non-data adaptive algorithms are Discrete Fourier Transform (DFT), Piecewise Aggregate Approximation (PAA), DCT (Discrete Cosine Transform), or Wavelets [20].

In **data adaptive** representations, the parameters vary depending on the available data. In the literature, we can find some well-known methods for data-adaptive representation, such as Symbolic Aggregate Approximation, Piecewise Linear Approximation, or Singular Value Decomposition [20]

The **model-based** approaches assume that the observed time series was created based on the basic model. The aim is to find the parameters of such a model as a representation. Two time series are then considered similar if an identical set of parameters can model them. The model can be a Hidden Markov, statistical, or even deep learning one [20].

### D. Data Scaling

For the scaling process, we propose selecting from three of the most famous and standard techniques: **normalization**, **standardization**, and **robust scaling**. Readers are referred to [21] for more detailed explanations of these scaling methods and how to select the right one based on the data distribution and the applications.

### E. Attention-based Bidirectional Long Short Term Memory Autoencoder as the Anomaly Detection Engine

As mentioned above, several detection models can detect collective anomalies after the segmentation, representation, and scaling process. Some examples are the One-Class Support Vector Machine, Isolation Forest, or AutoEncoder. However, we recommend using an Attention-based Bidirectional Long Short Term Memory Autoencoder as the anomaly detection engine. The previous works [22]–[25] also inspire this recommendation. The authors have proved the efficiency and robustness of LSTM- and Bidirectional LSTM- Autoencoder for the anomaly detection problem. Figure 3 illustrates a simplified structure of an Attention-based Bidirectional Long Short Term Memory Autoencoder.

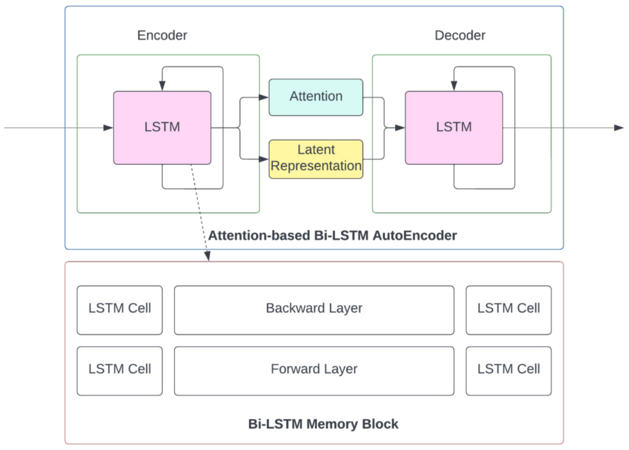

Fig. 3. Attention-based Bidirectional Long Short Term Memory Autoencoder

Because of the limitation of the pages, we will not describe the network in detail. Instead, readers, who are interested in this network, are referred to [25]–[27] for more information.

## IV. Experiments and Results

This section describes the dataset, accuracy measurement, and the results of the experiments.

### A. Dataset Description and Experiment Settings

The data used for the experiments in this article is the S5 dataset, provided by Yahoo [28]. This is a labeled benchmark dataset for anomaly detection. We compared the above-mentioned unsupervised methods based on their performance with this dataset. Therefore, it is essential to mention that the data labels are only used for the performance evaluation and not for the model training process.

The time series dataset represents the traffic of Yahoo services. The anomalies were labelled by experts. This dataset consists of 67 different time series. Each of them has 1400 data points, which were recorded hourly. About $1.9\%$ of the data are anomalies. The dataset is divided into training and test sets where $70\%$ of the data are used for training and $30\%$ for testing. The training set does not contain any abnormal sub-sequence. Figure 4 visualizes a time series with collective anomalies colored red.

### B. Accuracy Measurement

Because we have the labeled anomalies in the test set, AUC can be used to evaluate the framework's performance.

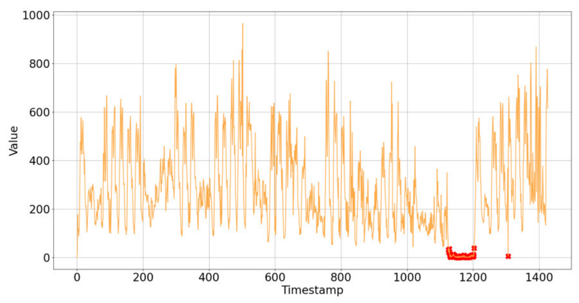

Fig. 4. A Time series with collective anomalies

AUC is the abbreviation of "Area under the ROC (Receiver Operating Characteristic) curve." That is, AUC represents the entire two-dimensional area under the ROC curve. ROC curve is a diagram showing the performance of a detection model at all values of thresholds. This curve illustrates two parameters: true positive rate (TPR) and false positive rate (FPR). The true positive rate is also known as the recall.

## C. Results

In this part of the section, the results of the experiments are discussed. After the segmentation process, which is mandatory, the optimal length of a sub-sequence is experimentally set to 4. The most suitable segmentation method for this dataset is the sliding windows algorithm. Because the window size is tiny, the non-data adaptive method was applied for the representation process. The transformed vectors are at the end scaled with a robust scaler. The experimental results show that all four main processes of the framework are essential for high detection accuracy. Missing one of them will lead to lower performance.

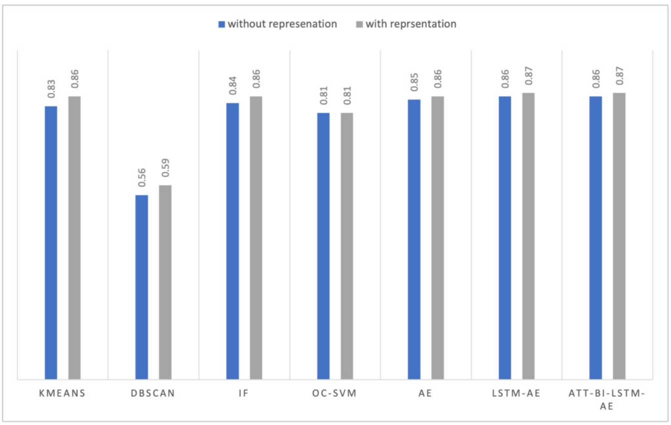

Fig. 5. Importance of the representation process

Figure 5 illustrates the importance of the representation process. The figure shows that the accuracy of six models (out of seven) is improved while applying the representation process, while the accuracy of the last one remains the same. Another critical remark is, together with LSTM AutoEncoder, Attention Bidirectional Long Short-Term Memory Autoencoder outperformed other detection models in both cases, with or without the time series representation process.

Figure 6 visualizes the performance ace of the udCATS framework with different detection models.

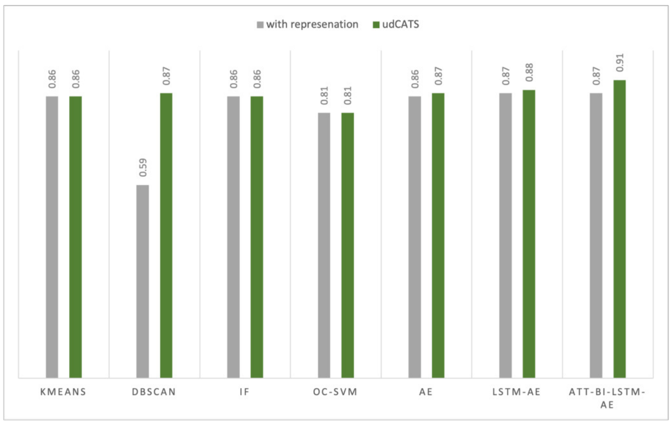

Fig. 6. Performance of udCATS Framework

From the graphic, it is crucial to observe that the scaling process of the comprehensive udCATS framework improved the accuracy of five detection models. The remaining two models performed at the same level. Besides, the udCATS framework with Attention-based Bidirectional Long Short Term Memory Autoencoder as the anomaly detection engine received the highest accuracy, represented by the AUC values. To obtain the best results, the confidence interval of the detection model is predetermined with a value of 0.95.

Table I shows the averaging AUCs of the models in different settings, while figure 7 illustrates the box plot of the udCATS framework's AUCs over the whole dataset. Besides the mean of the AUCs, which is 0.91, the box plot also shows their median. The median is very high, around 0.97. The box plot is short, which means the udCATS framework performs with a high level of agreement over the whole data set of 67 time series.

TABLE I
EXPERIMENTAL RESULTS

| **Detection Model** | **Without TS Representation** | **With TS Representation** | **udCATS** |
|---|---|---|---|
| K-Means | 0.83 | 0.86 | 0.86 |
| DBSCAN | 0.56 | 0.59 | 0.87 |
| IF | 0.84 | 0.86 | 0.86 |
| OC-SVM | 0.81 | 0.81 | 0.81 |
| AutoEncoder | 0.85 | 0.86 | 0.87 |
| LSTM AE | 0.87 | 0.86 | 0.88 |
| Att-Bi-LSTM-AE | 0.87 | 0.86 | **0.91** |

## V. Conclusion and Outlook

In this work, we provided two main contributions. Firstly, we experimentally demonstrated that an Attention-based Bidi-

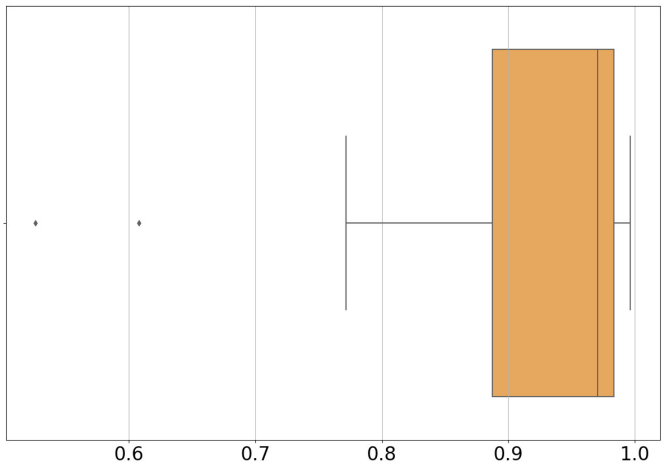

Fig. 7. Performance of udCATS Framework with Att-Bi-LSTM-AE

rectional LTSM Autoencoder could handle the collective anomaly detection of a time series. Secondly, and most importantly, we proposed a comprehensive framework, called udCATS, for solving the problem, which contains four main selecting processes: time series segmentation, representation, data scaling, and anomaly detection. To the best of our knowledge, this is the first comprehensive framework to handle this problem. The experimental results show that the Attention-based Bidirectional LTSM Autoencoder model performed better than the other detection models. Using it within the udCATS framework significantly improved the detection accuracy.

The following steps will assess the framework with more benchmark data sets. First, this would guide to an improvement of the framework architecture. Afterward, we will extend the selection processes with other methods and try to find a method to implement these processes to work fully automatically. Last but not least, we could combine the loss function of the four individual processes into one total loss function. The idea is to develop an end-to-end training process that improves accuracy.

## References

[1] Judith D Singer and John B Willett. It's about time: Using discrete-time survival analysis to study duration and the timing of events. *Journal of educational statistics*, 18(2):155–195, 1993.

[2] Douglas M Hawkins. *Identification of outliers*, volume 11. Springer, 1980.

[3] Mohammad Braei and Sebastian Wagner. Anomaly detection in univariate time-series: A survey on the state-of-the-art. *arXiv preprint arXiv:2004.00433*, 2020.

[4] Andrew A Cook, Göksel Mısırlı, and Zhong Fan. Anomaly detection for iot time-series data: A survey. *IEEE Internet of Things Journal*, 7(7):6481–6494, 2019.

[5] John Cristian Borges Gamboa. Deep learning for time-series analysis. *arXiv preprint arXiv:1701.01887*, 2017.

[6] Mohsin Munir, Shoaib Ahmed Siddiqui, Andreas Dengel, and Sheraz Ahmed. Deepant: A deep learning approach for unsupervised anomaly detection in time series. *Ieee Access*, 7:1991–2005, 2018.

[7] Loïc Bontemps, Van Loi Cao, James McDermott, and Nhien-An Le-Khac. Collective anomaly detection based on long short-term memory recurrent neural networks. In *International conference on future data and security engineering*, pages 141–152. Springer, 2016.

[8] Eamonn Keogh and Jessica Lin. Clustering of time-series subsequences is meaningless: implications for previous and future research. *Knowledge and information systems*, 8(2):154–177, 2005.

[9] Mete Çelik, Filiz Dadaşer-Çelik, and Ahmet Şakir Dokuz. Anomaly detection in temperature data using dbscan algorithm. In *2011 international symposium on innovations in intelligent systems and applications*, pages 91–95. IEEE, 2011.

[10] Wentai Wu, Ligang He, and Weiwei Lin. Local trend inconsistency: a prediction-driven approach to unsupervised anomaly detection in multi-seasonal time series. *arXiv preprint arXiv:1908.01146*, 2019.

[11] Mayu Sakurada and Takehisa Yairi. Anomaly detection using autoencoders with nonlinear dimensionality reduction. In *Proceedings of the MLSDA 2014 2nd workshop on machine learning for sensory data analysis*, pages 4–11, 2014.

[12] Aristidis Likas, Nikos Vlassis, and Jakob J Verbeek. The global k-means clustering algorithm. *Pattern recognition*, 36(2):451–461, 2003.

[13] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pages 226–231, 1996.

[14] Erich Schubert, Jörg Sander, Martin Ester, Hans Peter Kriegel, and Xiaowei Xu. Dbscan revisited, revisited: why and how you should (still) use dbscan. *ACM Transactions on Database Systems (TODS)*, 42(3):1–21, 2017.

[15] Seyedjamal Zolhavarieh, Saeed Aghabozorgi, and Ying Wah Teh. A review of subsequence time series clustering. *The Scientific World Journal*, 2014, 2014.

[16] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[17] Stephen D. Bay, Dennis F. Kibler, Michael J. Pazzani, and Padhraic Smyth. UCI machine learning repository, 1999.

[18] Miodrag Lovrić, Marina Milanović, and Milan Stamenković. Algoritmic methods for segmentation of time series: An overview. *Journal of Contemporary Economic and Business Issues*, 1(1):31–53, 2014.

[19] Xiaoyue Wang, Abdullah Mueen, Hui Ding, Goce Trajcevski, Peter Scheuermann, and Eamonn Keogh. Experimental comparison of representation methods and distance measures for time series data. *Data Mining and Knowledge Discovery*, 26(2):275–309, 2013.

[20] Chotirat Ratanamahatana, Eamonn Keogh, Anthony J Bagnall, and Stefano Lonardi. A novel bit level time series representation with implication of similarity search and clustering. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 771–777. Springer, 2005.

[21] Pallavi Pandey and Avinash Navlani. Feature scaling: Minmax, standard and robust scaler, Nov 2020.

[22] Sanket Mishra, Varad Kshirsagar, Rohit Dwivedula, and Chittaranjan Hota. Attention-based bi-lstm for anomaly detection on time-series data. In *International Conference on Artificial Neural Networks*, pages 129–140. Springer, 2021.

[23] Mahmoud Said Elsayed, Nhien-An Le-Khac, Soumyabrata Dev, and Anca Delia Jurcut. Network anomaly detection using lstm based autoencoder. In *Proceedings of the 16th ACM Symposium on QoS and Security for Wireless and Mobile Networks*, pages 37–45, 2020.

[24] HD Nguyen, Kim Phuc Tran, Sébastien Thomassey, and Moez Hamad. 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.

[25] Ashima Chawla, Paul Jacob, Brian Lee, and Sheila Fallon. Bidirectional lstm autoencoder for sequence based anomaly detection in cyber security. *International Journal of Simulation–Systems, Science & Technology*, 2019.

[26] Ariyo Oluwasanmi, Muhammad Umar Aftab, Edward Baagyere, Zhiguang Qin, Muhammad Ahmad, and Manuel Mazzara. Attention autoencoder for generative latent representational learning in anomaly detection. *Sensors*, 22(1):123, 2021.

[27] Jing Wang, Guigen Nie, Shengjun Gao, Shuguang Wu, Haiyang Li, and Xiaobing Ren. Landslide deformation prediction based on a gnss time series analysis and recurrent neural network model. *Remote Sensing*, 13(6):1055, 2021.

[28] N Laptev and S Amizadeh. Yahoo anomaly detection dataset s5. *URL http://webscope. sandbox. yahoo. com/catalog. php*, 2015.