Anomaly detection models for time series

Alexander Fandos Jahrsetz
Master's degree in artificial intelligence research
Universitat Internacional Menéndez Pelayo
Madrid, Spain
alexanderfandos@posgrado.uimp.es

***Abstract*—Anomaly detection for time series is a task with increasing importance in the recent years. This task refers to the discovery of points or subsequences in time series that are not expected in such time series. This work focuses in the analysis of the most recent techniques. The report is divided in three main parts. In the first one time series and anomalies are introduced. Afterward anomaly detection models are introduced and some selected models are explain in depth. Finally, those selected models are tested and their performance discussed. The final goal of this work is to make some of the latest models implementations available for public use.**

***Index Terms*—Time series, anomalies, outliers, discords, machine learning**

I. Introduction

Every year an immense amount of data is being created and consumed. This amount increases day by day. For example, according to Statista in 2021 79ZB were consumed, whereas in 2022 it is estimated that 97ZB will be consumed [36]. Business, governments, and many others take advantage of this data to improve their products and systems [37] [38]. However, analyzing the complete data manually is impossible. Machine learning models have been developed to automate these tasks. Prediction, detection and segmentation are some of the tasks with the most mature models. Anomaly detection on the other hand is a much less explored topic, and has received a lot of attention in the recent years as seen in figure 2. Data is expected to be inside a given normality. However in some cases the data differs from normality. These data points are anomalous and may contain relevant information such as disease in blood analysis or fraudulent movements in a bank account to name some examples. In most cases the fed data comes in the form of independent vectors or matrices. But it may also come in the form of time series. This adds a layer of complexity as temporal correlation has to be taken into account. Anomaly detection in time series had almost zero attention in the early 2000s. However, the number of publications has increased significantly since then, even though compared to general anomaly detection research articles they are still low. As this is still an unexplored field there are still few available models for public use. Therefore this work's aim is to study the most recent approaches that do not have public available implementations, implement, compare and publish them on GitHub.

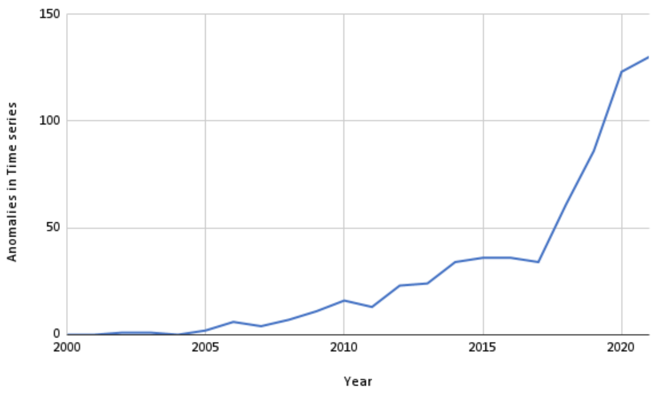

Fig. 1. Number of articles with the tags "time series" and "anomalies" or "discords" or "outliers" published in IEEE by year.

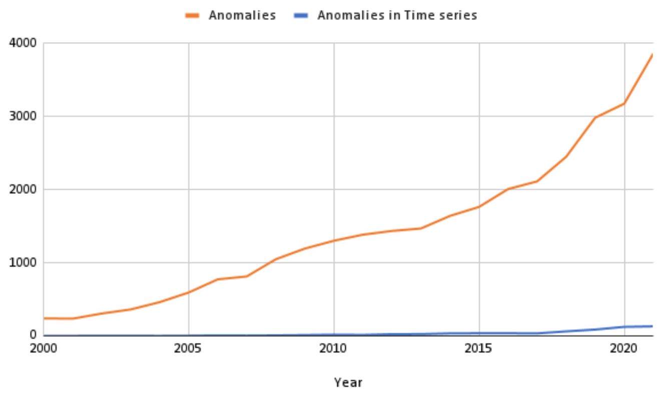

Fig. 2. Number of articles with the tags "anomalies" or "discords" or "outliers" published in IEEE by year compared to those of figure 1

II. Definitions

A. Time series

A time series is a series of values or data obtained in successive times. They are indexed (with timestamps in most cases) and each point is often taken in equally spaced points in time. However, the latter is not mandatory, as the sampling may be varying or some data points might be missing.

Time series can be categorized by the nature of their data.

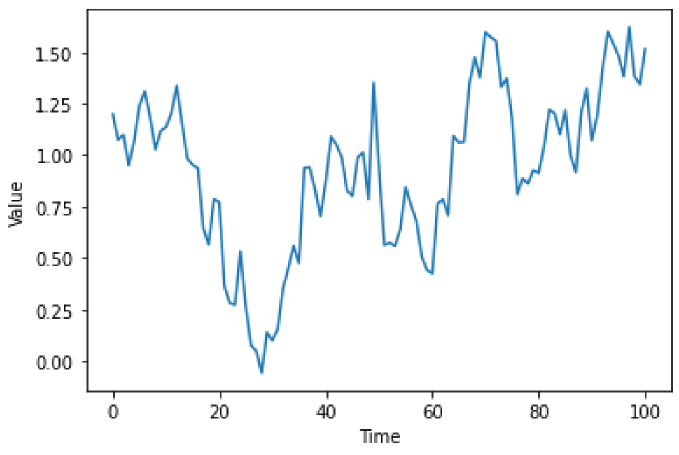

Fig. 3. An example of univariate time series.

1) Univariate: The most common type of time series in the literature. Each data point consists of a scalar value. For example the recording of temperatures over time of a single thermometer is a univariate time series.

2) Multivariate: In multivariate time series each data point is a 1 or higher dimensional tensor. Sometimes multivariate time series are considered as a group of univariate time series occurring at the same period of time. For example the recording of temperatures over time of two or more thermometers is a univariate time series. Another example is a video, where each frame is a data point of the time series.

B. Anomalies

Anomalies have been called by many names in literature. Some call it anomalies, others outlier, discordant observations, discords, exceptions, aberrations, surprises, peculiarities or contaminants [1]. Hawkins [2] used the word outlier and defined it as:

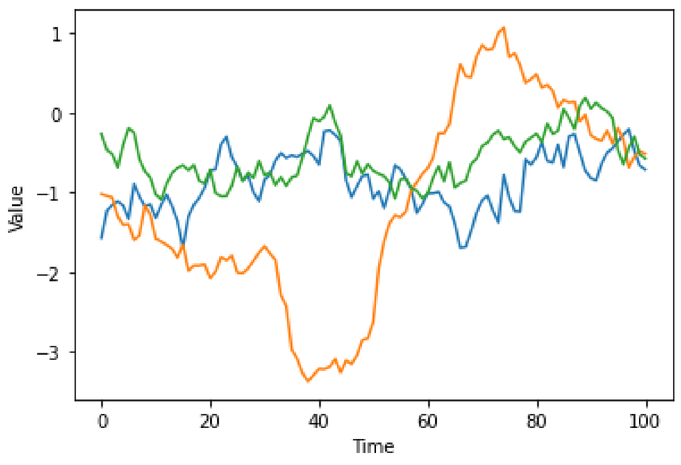

Fig. 4. An example of multivariate time series.

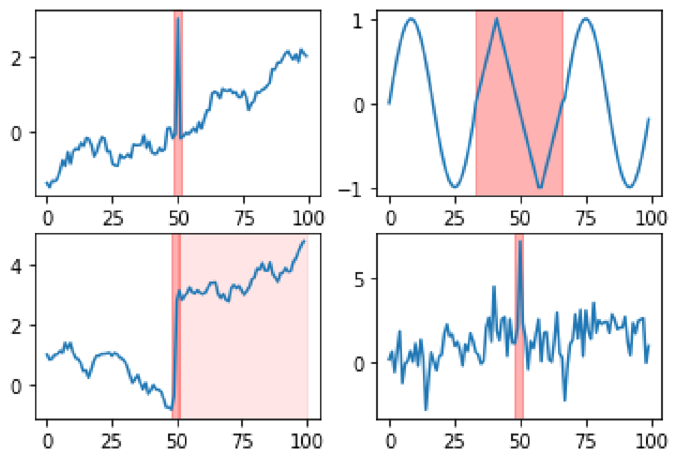

Fig. 5. Different types of anomalies. Top left: a peak that is not expected. Top right: sudden triangular waveform instead of sine wave. Bottom left: sudden change of mean value. Bottom right: peak inside noisy time series.

"An observation which deviates so much from other observations as to arouse suspicions that it was generated by a different mechanism"

In other words, it is a data point or data points that behave differently than the rest.

Anomalies can take many forms. For example a point in data that has a much higher value than the rest of points is an anomaly. A sinusoidal signal that suddenly and momentarily changes to a triangular signal (even though with the same amplitude) is an anomaly. Another example is a time series in which the mean value rises abruptly but maintains the new mean value. The point in which the mean value rises is an anomaly. But the whole time series starting from the new mean value could also be identified as an anomaly. The points in a time series with a lot of noise that deviate a lot from the mean may not be considered anomalies, as it is an expected behavior. Identifying and classifying anomalies is a hard task.

To make things simpler and be able to categorize the anomaly detection models the following categorization is proposed:

1) Dimension: Time series can be univariate and multivariate. Anomalies too. In univariate time series only univariate anomalies can appear. In multivariate time series, both types can appear. If only one dimension has an unexpected behavior it would be an univariate anomaly. If all single dimensions behave properly, but the combination of them all is unexpected, it is a multivariate anomaly. For example, in figure 7 a multivariate anomaly is depicted. Each dimension can take the value 0 or 1 and are usually inverted. Therefore, if both dimensions take the same value, it's an anomaly.

2) Temporal length: A single point can be anomalous. In other cases a succession of points are anomalous. The former is called point anomaly and the latter subsequence anomaly.

III. Models

A search for surveys about anomaly detection models in time series has been carried out. As the aim is to find the

TABLE I
MODELS

Technique	Author	Year	Type	Technique	Point/ Subsequence	Dimension	Implementation available
AAMP [3]	Akbarinia and Cloez	2019	Unsupervised	Matrix Profiling	Subsequence	Univariate	No
ACAMP [3]	Akbarinia and Cloez	2019	Unsupervised	Matrix Profiling	Subsequence	Univariate	No
Cross-correlation dimension reduction [15]	Lu et al.	2018	Unsupervised	Dimension reduction	Point	Multivariate	No
DeepAnT [4]	Munir et al.	2019	Semi Supervised	Prediction	Subsequence	Multivariate	No
Density-based outliers with sliding windows [11]	Ishimtsev et al.	2017	Semi Supervised	Density-based	Point	Univariate	Yes [29]
Dynamic clustering [18]	Wang et al.	2018	Unsupervised	Clustering	Subsequence	Univariate	No
EnrichedCNNAE [5]	Kieu et al.	2018	Semi Supervised	AutoEncoder	Subsequence	Multivariate	No
GMM [6]	Reddy et al.	2017	Semi Supervised	Estimation	Point	Univariate	No
HOT-SAX with clustering categorical data [17]	Chau et al.	2018	Unsupervised	Density-based	Subsequence	Univariate	No
LRRDS [7]	Hu et al.	2019	Unsupervised	Dimension reduction/d Density-based	Subsequence	Multivariate	No
Multivariate LSTM with variable independent outlier search [16]	Hundman et al.	2018	Semi Supervised	Dimension reduction/ Prediction	Point	Multivariate	Yes [32]
Non-parametric model [19]	Zhou et al.	2018	Semi Supervised	Prediction	Point	Multivariate	No
PAPRRW [8]	Ren et al.	2017	Semi Supervised	Density-based	Subsequence	Univariate	No
Resiuduals using STL decomposition [12]	Hochenbaum et al.	2017	Semi Supervised	Estimation	Point	Univariate	Yes [30]
SCRIMP++ [13]	Zhu et al.	2018	Unsupervised	Matrix Profiling	Subsequence	Univariate	No
Semiparametric outlier detection [14]	Holešovský et al.	2018	Unsupervised	Estimation	Point	Univariate	Yes [31]
swARIMA [9]	Zhou et al.	2018	Unsupervised	Prediction	Point	Univariate	No
VAE with GRU [20]	Su et al.	2019	Semi Supervised	AutoEncoder	Point	Multivariate	Yes [33]
WaveNet [10]	Borovykh et al.	2017	Semi Supervised	Prediction	Point/Subsequence	Univariate	Yes [28]

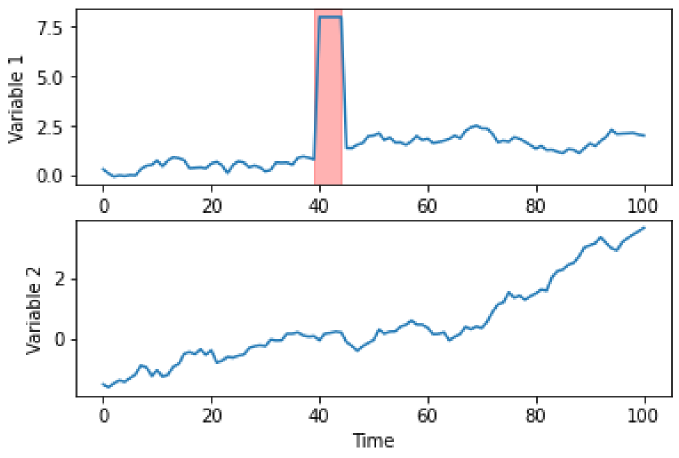

Fig. 6. An example of univariate anomaly in a multivariate time series. Only variable 1 has an anomalous behaviour.

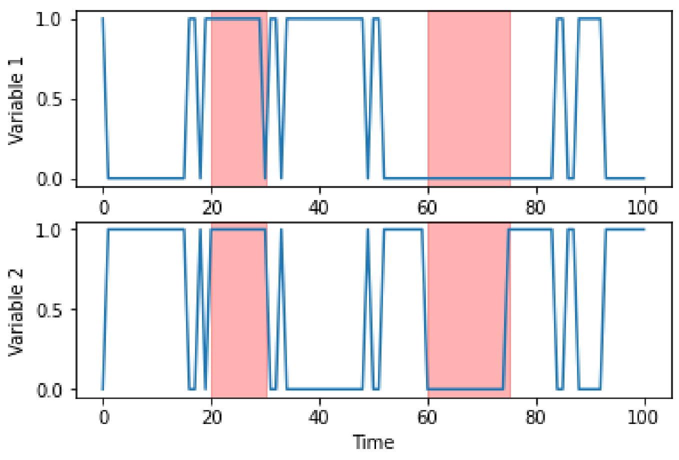

Fig. 7. An example of multivariate anomaly in a time series. Variable 2 works as the inverse of Variable 1, therefore both variables can't have the same value at the same time.

most recent anomaly detection models only surveys published after 2019 have been considered. Several reviews have been found. One [24] just presents a taxonomy for classification of anomaly detection and describes briefly each type of detection technique. [25] is the most extensive survey found. It also presents a taxonomy for anomaly detection methods in time series and presents and explains different models for each category. [26] presents and explains in great detail detection techniques for univariate time-series. [27] is a shorter review, but is the only one that introduces Matrix Profiling techniques, which is a promising approach for anomaly detection.

The extracted methods are listed in table I. Only those published starting from 2017 have been considered. [25] also mentions 3 other models (DTW clustering [21], Shapelets [22] and Agglomerative hierarchical clustering [23]) that detect anomalous time series. These types of models do not label points or subsequences as anomalous, but whole time series as anomalous or not. They are not included in this list, as such techniques are not the aim of this work.

Models can be supervised, semi-supervised and unsupervised. Supervised models need training data with anomaly labels and will try to learn what an anomaly is based on those labels. These type of techniques are usually not popular in anomaly detection as it can been seen that no recent model uses such approach. Semi supervised models also need training data, however it does not need to be labeled. This training data is usually used to learn the concept of normality. If new data is different than the learned normality, its considered an anomaly. Training data for semi supervised techniques need to have the lowest number of anomalies as possible, as in the opposite case the model would consider anomalies as normal data. Supervised and semi-supervised models usually take a long time for training, but are fast at the inferring process. Unsupervised models do not need training, however are usually slow at the inferring process, as they will try to learn what normal and anomalous data looks like while inferring.

There are all sorts of used techniques in the latest publications. Prediction based techniques are popular and easy to comprehend. Such techniques try to predict a point in time with its preceding data point and then compare it to the real point. If predicted and real points differ over a threshold, it is considered an anomaly. This can be applied to univariate time series (in which the absolute difference can be used) or multivariate time series. In the latter case some sort of difference needs to be calculated, for example the Euclidean or Manhattan distance.

Estimation based techniques are similar to prediction based techniques, but instead of relying only on previous data they also rely on the current and / or the following data points. For example, using the previous, current, and next datapoint to obtain a mean value and compare it to the current value would be an estimation based technique.

Density-based techniques rely on the idea that anomalous points or subsequences do not have many neighboring points or subsequences. In this sense these techniques search for points or subsequences that are different from the rest.

Matrix Profiling is a technique to compare 2 time series. It uses a sliding window of size m to iterate through the first time-series and returns for each iteration distance to the closest subsequence of size m in the second time series and its index. Those values are stored in the distance profile vector and the profile index vector. If the first and second time series are the same, the distance profile vector can be used to detect outliers as high values will show subsequences that are very different to the rest of the time series.

Auto-encoder based techniques train an auto-encoder with training data. Training data needs to have a low amount of anomalies, as this type of technique relies on the idea that the autoencoder will not be able to reconstruct the data properly if it is an anomaly. They work for univariate and multivariate time series. For multivariate time series they can encode each point. However, in univariate time series a single point can't be encoded, therefore, several points are encoded at once.

Finally, dimension reduction based techniques work only in multivariate time series. They reduce the dimensionality of time series to obtain a univariate time series in which a univariate model can be applied.

Not all models have been implemented. Some, like WaveNet, density-based outliers with sliding windows, resiuduals using STL decomposition, semiparametric outlier detection and multivariate LSTM with variable independent outlier search, already have a public implementation. SCRIMP++ has been discarded as ACAMP is an improved version of it. Non-parametric model is a technique designed to solve a specific problem, so it was also discarded. Finally, cross-correlation dimension reduction, HOT-SAX with clustering categorical data and dynamic-clustering do not provide enough details on its article to implement the model properly.

A. AAMP

AAMP is a matrix profiling based technique. As explained before, these techniques find for each subsequence the subsequence that is the most similar to it. This an extremely time consuming process if done by brute force. Matrix profiling based techniques try to improve the time complexity of this search. AAMP is such a technique and is faster than the previous published matrix profiling techniques. Matrix profiling techniques usually compute the Z-normalized euclidean distance, however AAMP computes the raw euclidean distance. According to the author, in some cases magnitude incorporates valuable information for anomaly detection which is discarded in other matrix profiling techniques.

By brute force the algorithm would compute the Euclidean distance for each subsequence pair from scratch. AAMP computes the euclidean distance step by step. Assume 2 subsequences S_i and S_j. S_i starts at time i and S_j starts at time j and the both have a length of m. The euclidean distance between S_i and S_j is $d_{i,j}$ is defined by:

$$d_{i,j} = \sqrt{\sum_{k=0}^{m-1} (t_{i+k} - t_{j+k})^2} \tag{1}$$

Where t_x is the x-th value of the time series.

Now, to obtain distance $d_{i+1,j+1}$ equation 1 is not used which would need to iterate again over m*2 elements. Instead the following trick can be used:

$$d_{i+1,j+1} = \sqrt{d_{i,j} - (t_i - t_j)^2 + (t_{i+m} - t_{j+m})^2} \tag{2}$$

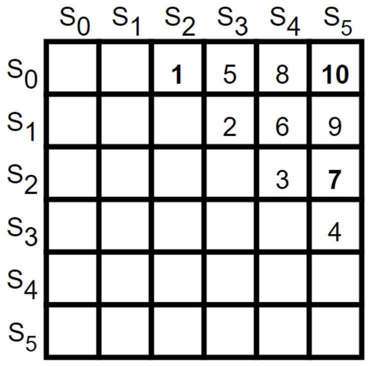

Fig. 8. An example of AAMP's iteration process with k = 2, l - m = 5. Each cell represent a comparison of two subsequences. The numbers inside each cell represent their order.

This way, the computation of distance of all pairs of subsequences is much faster.

The way in which AAMP iterates over subsequences is also particular. First of all it takes into account a minimal distance k for comparison. This makes a lot of sense as two subsequences one timestamp a part will almost always be very similar and thus even anomalous subsquences would return low difference values. Assume the time series has a length of l.

First subsequence S_0 starting from timestamp 0 is compared to subsequence S_k starting from timestamp k. The most common way of iterating would be comparing next S_0 to S_{k+1}, however, the before mentioned trick would not work. Therefore S_1 is compared to S_{k+1} and so on till S_{l-k-m} is compared to S_{l-m}.

Next, the algorithm starts over comparing S_0 but this time to subsequence S_{k+1} and so as before. The last iteration will compare S_0 to S_{l-m}. This create a diagonal iteration as it can been seen in figure 8

B. ACAMP

ACAMP was presented in the same publication as AAMP. ACAMP is based on the same concept as AAMP, but uses Z-normalized euclidean distance. Equation 2 cannot be used. Instead the following equation is used to compute the Z-normalized euclidean distance:

$$dz_{i,j} = \sqrt{2m\left(1 - \frac{C_{i,j} - \frac{1}{m}sA_i sB_j}{\sqrt{(ssA_i - \frac{1}{m}sA_i^2)(ssB_j - \frac{1}{m}sB_j^2)}}\right)} \tag{3}$$

where
$sA_i = \sum_{l=0}^{m-1} t_{i+l}$ is the sum of all values in S_i
$sB_j = \sum_{l=0}^{m-1} t_{j+l}$ is the sum of all values in S_j
$ssA_i = \sum_{l=0}^{m-1} t_{i+l}^2$ is the sum of squares of all values in S_i
$ssB_j = \sum_{l=0}^{m-1} t_{j+l}^2$ is the sum of squares of all values in S_j
$C_{i,j} = \sum_{l=0}^{m-1} t_{j+l} \times t_{j+l}$ is the product of all values in S_i and S_j

The way these values are updated for every step is:

$$sA_i = sA_{i-1} - t_{i-1} + t_{i+m-1} \tag{4}$$

$$sB_j = sB_{j-1} - t_{j-1} + t_{j+m-1} \tag{5}$$

$$ssA_i = ssA_{i-1} - t_{i-1}^2 + t_{i+m-1}^2 \tag{6}$$

$$ssB_j = ssB_{j-1} - t_{j-1}^2 + t_{j+m-1}^2 \tag{7}$$

$$C_{i,j} = C_{i-1,j-1} - t_{i-1} \times t_{j-1} + t_{i+m-1} \times t_{j+m-1} \tag{8}$$

C. DeepAnT

DeepAnT is a prediction based anomaly detection technique. DeepAnT's prediction is based on convolutional neural networks. With a sliding window the preceding data points are directly fed into the neural network to obtain a prediction. DeepAnT is supposed not to be data hungry and should work with few examples according to its author. Its architecture is rather short. It consists of a convolutional layer, followed by a max pooling layer, another convolutional layer, another max pooling layer and finally a dense fully connected layer. Each convolutional layer consists of 32 1-dimensional filters, followed by a ReLU activation function. The loss function it optimizes is the mean absolute error.

$$MAE = \frac{1}{n}\sum_{i=1}^{n} abs(y_i - \hat{y}_i) \tag{9}$$

D. EnrichedCCNAE

The problem with multivariate autoencoders used to detect anomalies in time series is that they do not take into account temporal correlation between consecutive data points. To overcome this limitation the authors propose enriching the data to include temporal information. This is done with a sliding window of size m, that moves around the time-series T in m/2 steps. For each subsequence it computes its norm and the difference to the norm of the previous subsequence variable by variable. To this new time series T' a new sliding window of length n and step n/2 is applied. The following properties are obtained for the norms and the difference of norms: Mean, minimum, maximum, 25%-quartile, 50%-quartile, 75% quartile, standard deviation and peak to peak. Again, this is done variable by variable obtaining time series T". So at the end, if the original time series T had for example 3 variables for each data point, T" has a 16x3 data matrix for each data point. These matrices contain enough temporal information for the autoencoder to work properly and detect anomalies in the original time series T.

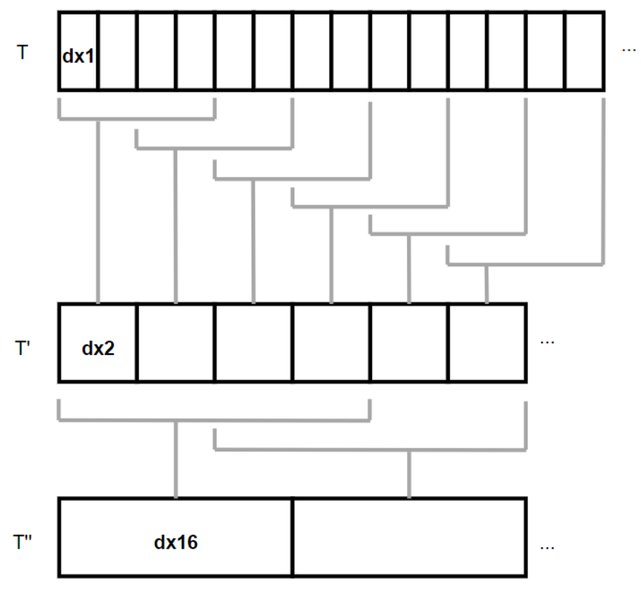

Fig. 9. An example of how the enriched time series T" is constructed in EnrichedCCNAE

E. GMM

GMM stands for Gaussian Mixture Model. GMMs can be used as a clustering algorithm but they can be also used to estimate how probable a new value belongs to a cluster. This can be exploited to detect anomalies. If the probability of a new value to belong to the existing data is low it is an anomaly. However, modeling the whole Time Series with a single GMM would not return trustworthy results, as for example, the power consumption at night has a completely different behavior than the power consumption at midday. The authors have adapted GMMs especially for seasonal time series. They divide each period in n bins, and use the bins from all periods to train n GMMs, one for each bin. They also scale the GMMs output to a range from 0 to 10, in which 0 is normal data and 10 an extreme outlier.

F. LRRDS

LRRDS is able to detect variable-length anomalous subsequences. The subsequence segmentation is done with the use of recurrence plots. A recurrence plot shows for all points in the time series what other points of the same time series have a similar phase space. This involves comparing subsequences with subsequences. To accelerate this process the authors compress the time series of length l by a compression rate c. This is done dividing the time series in subsequences of length c and replacing the subsequence by their mean value, which will create a timeseries of length l/c, like in figure 10.

The authors have found out that comparing subsequences of length 1 is enough for their purpose, which is subsequence segmentation. So actually, this process does not involve comparing subsequences, but points.

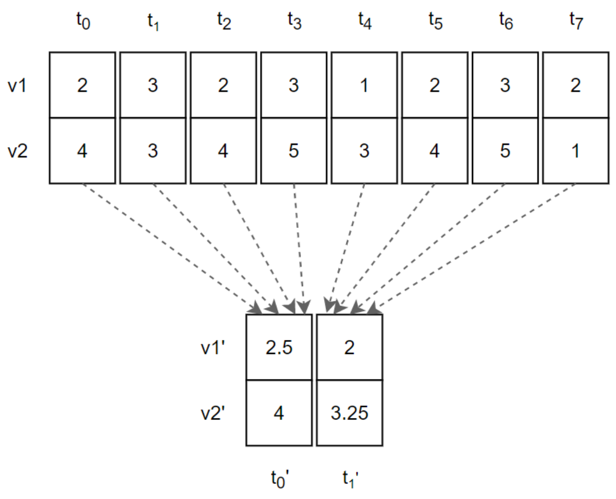

Fig. 10. An example of compression with compression factor c = 4

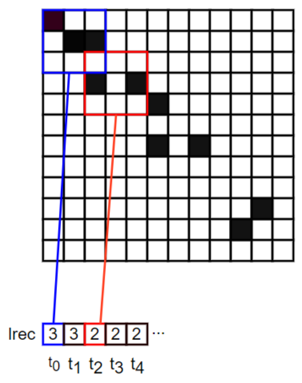

Fig. 11. An example of how the first and third value of lrec are computed with a windows size of 3x3

Before comparing, the data is normalized to a range from 1 to 2. 2 comparisons are done for each pair of points, one with the Euclidean distance and another with Bhattacharyya distance.

$$ED = \sqrt{\sum_{h=1}^{d} (e_{ih} - e_{jh})^2} \tag{10}$$

$$EB = \sqrt{1 - \frac{\sum_{h=1}^{d} (e_{ih} e_{jh})}{\sum_{h=1}^{d} (e_{ih}) \sum_{h=1}^{d} (e_{jh})}} \tag{11}$$

Where e_{ih} and e_{ih} represent the i-th and j-th timestamp of the h-th variable.

Both results are binarized with a threshold and combined with a logical "and" operation. This creates a l/c x l/c matrix. An nxn sliding window will be used to iterate through the diagonal of the matrix. Therefore, not all values of this matrix need to be calculated, but only those that are at a Manhattan distance of n from the diagonal of the matrix.

The sum of all points in each window is obtained and stored in a vector of length l/c - n called lrec. This vector is first binarized into lrec'. Every value different to 0 is turned into 1. Now each time lrec' changes value, a new subsequence starts.

For each subsequence the length and its mean value in lrec is extracted. Each subsequence's length and mean are compared to each other with the same process (logical "and" operation of binarized Euclidean and Bhattacharyya distance) as before generating a new matrix of size sxs where s is the number of subsequences. The number of values equal to 1 is counted for each row and fed into a k-means model with k=2. This will classify subsequences into normal and anomalous.

G. PAPRRW

The PAPRRW [8] technique segments the time series in fixed length subsequences. For each subsequence n equal probability sub value spaces are calculated. This means that if any new value is sampled from the value distribution of that subsequence, it has equal probability of falling into any of those n sub spaces. For each sub value space the number of points inside each space, their mean and variance are calculated. With these 3 properties, each subsequence is characterized by a 3xn matrix. With each subsequence's matrix, a similarity matrix is constructed. A random walk model is built, in which each node represents a subsequence and the similarity matrix is used as the relationship table of nodes. After some iterations a stable probability distribution is obtained. The lower the probability of a node/subsequence is, the more suspicious it is of being anomalous.

H. swARIMA

Using ARIMA to predict values and compare them to the real values is a popular technique to identify anomalous points and is great with global outliers. However it is not that great at detecting local outliers. Therefore the authors of swARIMA retrain the ARIMA model using a sliding window that moves forward in steps. Every time the ARIMA model is retrained, its p, d and q parameters are optimized. d is found by differentiating the selected sequence and checking if it passes the Dicker-Fuller test which checks that the sequence is stationary. The number of times it needed to be differentiated in order to to be stationary is d. p and q are obtained by training the model with different p and q values and selecting those that minimize the Akaike information criterion. In this way SwARIMA is able to learn the local behavior better and detect local anomalies.

I. Baseline models

To compare those models not only to themselves but also prior existing models, some already implemented models in Merlion have been tested. Those that could be applied on the used datasets without any transformation were selected. The selected models are:

- Autoencoder
- Deep autoencoding Gaussian mixture model for anomaly detection
- Isolation Forest
- Arima based anomaly detection
- Deep Point Anomaly Detector

IV. Performance evaluation

In this section all newly implemented models and baseline models will be compared. First the experimental setup will be described and then the results of the experimental evaluation will be presented and discussed.

A. Setup

The models were implemented in Python with the Merlion library [34]. Merlion is a library that offers a machine learning framework for time series. It offers methods for loading and transforming data, building and training forecasting and anomaly detection models, post-processing and evaluation. It is designed to work with both, univariate and multivariate time series. New anomaly (and also forecasting) models can be built easily around their base anomaly detection class. It also includes already built in models, which were used in this work as baseline models.

There exist other frameworks to work with time series, like for example Kats [35], which includes almost the same features as Merlion. However, Kats seems to have its own anomaly detection model and doesn't allow users to add new models. All other frameworks offer much less features than Merlion [34]. Therefore Merlion was selected as the framework to work with.

Merlion also offers easy access to several datasets specially recopilated for anomaly detection. These include all types of anomalies. 10 of these datasets have been handpicked for testing purpose with the following criteria:

- 8 univariate and 2 multivariate time series.
- Their anomalies are detectable by human eye and are describable, except for multivariate time series, which are all difficult to identify by bare eye.
- Time series types should be varied.
- Anomaly types should be varied.

The selected time series can be seen in figure 12 and their main feature and anomaly types are presented in table II.

Mutlivariate models will be tested with all datasets. However, univariate models will be tested only with univariate datasets.

Datasets have not been pre processed before applying to the model. Models have been tuned for each dataset to optimize F1-score. Recall, precision and F1-Score have been recorded. The training and testing time have also been recorded. Training time includes testing the training data, this is why techniques without a training phase have non-zero training times. Those experiments that took over 30 minutes to train or test were not finished. For recall, precision and F1-score mean they were not taken into account, however for time means, they are taking into account as a 60 minutes process.

It is expected that results will not be extremely high, as the existing labeling has been used. The existing labeling uses mostly variable length subsequence anomalies. However some of the models return only point anomalies, and will therefore

TABLE II
DATASETS

Merlion code	Short name	Univariate/ Multivariate	Seasonal	Seasonality	Other feature	Anomaly types
NAB(subset="artificialWithAnomaly")[1]	awa-1	Univariate	Yes	Daily	Artificial	Season max value decreases a lot.
NAB(subset="artificialWithAnomaly")[4]	awa-4	Univariate	Yes	Every 8 hours	Artificial, Binary	Increase in frequency of season.
NAB(subset="realAWSCloudwatch")[0]	rac-0	Univariate	Yes	Daily	Discrete Values	Unexpected values.
NAB(subset="realAdExchange")[1]	rae1-1	Univariate	Yes	Daily	Moving average	Shape of the seasons changes.
NAB(subset="realKnownCause")[3]	rkc-3	Univariate	No			Peaks are expected in the timeseries. However they should not keep in time, but some do.
NAB(subset="realKnownCause")[4]	rkc-4	Univariate	Yes	Daily and Weekely		Some days have concerning lower or higher values than their neighbors.
NAB(subset="realTraffic")[1]	rtr-1	Univariate	Yes	Daily	Missing timestamps	Succession of unexpected peak values.
NAB(subset="realTweets")[2]	rtw-2	Univariate	Yes	Daily	Moving max peaks in seasons	Some max peaks differ too much from their neighboring seasons.
SMD['machine-2-2][0]	smd-2-2-0	Multivariate	Yes	Daily		Different types of anomalies.
MSL()[0]	msl-0	Multivariate	No			Different types of anomalies. anomalies.

return a low F1-score as recall is also low. In other cases the labeling may set a whole period as anomalous and the model however just marks the anomalous part of the period as anomalous.

B. Results

AAMP and ACAMP perform extraordinarily well. Sometimes AAMP is better than ACAMP and other times ACAMP is better, however AAMP is generally better. It's also among the fastest models for small datasets. However, as its time complexity is quadratic it gets slow with growing datasets which makes them the slowest overally. They even couldn't handle the realKnownCause-3 dataset, as it was too big for it and took over 1 hour to process. ACAMP wasn't able to process the realTweets-2 database as well. PAPRRW and EnrichedCCNAE also return good results. EnrichedCCNAE needs a training phase and takes a while being trained. However in the prediction phase EnrichedCCNAE is extremely fast taking only 5.78 seconds in the worst case. PAPRRW however is toghether with AAMP and ACAMP one of the slowest models. It could be faster with less iterations in its RW process at the expense of poorer results. LRRDS sometimes returns very good results and other times extremely poor results. This is due to its ability to detect subsequences which fails often and on the other hand it's because the poor information used to describe the subsequences. It may also be due to its data compression, which makes LRRDS a fast model. EnrichedCCNAE and PAPRRW use much richer information to describe subsequences. LRRDS has also a lot of parameters that need precise tuning for each dataset. GMM, swARIMA and DeepANT detect anomalous points instead of subsequences. Labeling in Merlion works with anomalous subsequences. Therefore their recall results are expected to be much lower. Their results are more difficult to interpret. GMMs and SwARIMA will not work with high variance datasets, as they will label too many points as anomalous. Increasing their threshold increases precision but lowers recall fast. GMMs also do not work with binary data such as artificialWithAnomaly-4 dataset. GMM is very fast with training and prediction, however swARIMA is very slow as it needs to retrain its model even when predicting. DeepAnT is quite fast in its training phase for univariate datasets. However with multivariate datasets it slows down a lot. Its overall results are pretty low because of low recall.

Compared to the baselines models a great improvement can be observed. However, it has to be kept in mind that all baseline models are point anomaly detectors and thus, report worse results. Nonetheless even the new point anomaly models have generally better results. swARIMA is a great improvement from the base ARIMA detection model. The ARIMA model is not capable of detecting anomalies in most datasets, however swARIMA detects at least some true anomalies in most datasets. EnrichedCNNAE is also an extreme improvement from the base AutoEncoder model. This is because AutoEncoder does a poor job encoding temporal correlation. The only aspect in which the baseline models outperform the new models is in time complexity, even though they are just slightly faster.

In figure 13 F1-Score is plotted against Test time. Generally speaking it could be said that the better the model works, the slower it is.

Additionally a statistically analysis has been performed on

TABLE III
RECALL

	awa-1	awa-4	rac-0	rae-1	rkc-3	rk-4	rtr-1	rtw-2	smd-2-2-0	msl-0	mean
AAMP	**100%**	**100%**	88.55%	49.38%	-	54.68%	**100%**	65.09%			**79.67%**
ACAMP	**100%**	**100%**	**100%**	49.38%	-	77.87%	0%	-			71.21%
EnrichedCNNAE	2.97%	51.11%	62.43%	44.44%	6.46%	20%	53.91%	**83.42%**	16.66%	20.48%	36.19%
GMM	27.04%	-	2.48%	3.08%	32.51%	11.4%	1.38%	6.4%			12.04%
LRRDS	27.29%	0%	37.31%	12.96%	**96.06%**	62.02%	**100%**	72.31%	**43.84%**	**26.88%**	47.87%
PAPRRW	70.71%	44.66%	85.32%	**93.82%**	65.25%	**90.04%**	33.17%	24.67%			63.46%
swARIMA	41%	12.4%	64.92%	26.54%	0%		2.76%	30.19%			25.40%
DeepAnt	4.46%	26.79%	0.74%	4.93%	6.81%	4.63%	3.68%	2.25%	0.95%	3.43%	5.87%
AutoEncoder	2.24%	1.24%	2.48%	0%	2.58%	3.47%	0%	2.38%	0.81%	0.11%	1.53%
DAGM	2.48%	0.74%	0%	0%	0.82%	0.57%	0%	0%	0.74%	0.24%	0.56%
Isolation Forest	0.24%	0%	0.74%	3.7%	0.52%	0.19%	0%	0.81%	0.67%	0.19%	0.71%
Arima	0%	1.24%	0%	32.71%	0%	0%	0%	0.12%	-	-	4.26%
DeepPointAnomalyDetector	0%	0.99%	1.24%	0%	0%	0%	0%	3.95%	0%	0%	0.62%

TABLE IV
PRECISION

	awa-1	awa-4	rac-0	rae-1	rkc-3	rkc-4	rtr-1	rtw-2	smd2-2-0	msl-0	mean
AAMP	62%	68.42%	62.55%	**75.72%**	-	22.97%	37.67%	95.4%	-	-	**60.68%**
ACAMP	71.7%	68.42%	12.36%	68.96%	-	27.19%	0%	-	-	-	41.44%
EnrichedCNNAE	**100%**	32.85%	42.32%	40.44%	**100%**	4.86%	**82.97%**	51.49%	15.82%	12.7%	48.35%
GMM	74.65%	-	16.12%	22.72%	92.84%	47.041%	9.09%	53.68%	-	-	45.16%
LRRDS	20.1%	0%	**100%**	14.38%	17.95%	14.32%	48.22%	15.33%	13.95%	12.66%	25.69%
PAPRRW	100%	33.33%	43.41%	34.15%	63.28%	53.13%	33.96%	92.9%	-	-	56.77%
swARIMA	14%	**100%**	37.17%	11.49%	0%	-	10.71%	48.1%	-	-	31.64%
DeepAnt	2.13%	14.97%	23.07%	6.89%	45.66%	65.75%	47.05%	72%	**79.41%**	**25.82%**	38.28%
AutoEncoder	**100%**	**100%**	26.31%	0%	22.56%	37.89%	0%	27.73%	37.7%	4.61%	35.68%
DAGM	76.92%	**100%**	0%	0%	93.33%	**100%**	0%	0%	15.67%	12.25%	39.82%
Isolation Forest	4.54%	0%	21.42%	33.33%	18.75%	66.6%	0%	33.33%	38.77%	16.45%	23.32%
Arima	0%	19.23%	0%	12.55%	0%	0%	0%	**100%**	-	-	16.47%
DeepPointAnomalyDetector	0%	11.42%	12.5%	0%	0%	0%	0%	11.81%	0%	0%	3.57%

F1-scores. For this only result of univariate datasets have been used. Missing F1-scores have been artificially build. Those that are missing because of too high time processing have been replaced by the mean F1-score of their model. Those that are missing because the model cannot process the dataset have been replaced by 0.

First of all a Friedman Rank Sum test has been applied with all models to obtain each model's average rank and the p-value to reject the hypothesis that all models behave the same. The obtained p-value is 1.361e-8, which is extremely lower than the standard 0.05 rejecting p-value, so we can safely assume that all models have not the same behaviour. Once this is clear multiple post-hoc comparison tests have to be performed to see if there are groups of models that behave similarly. The selected test is the paired t-test. The average rank of each model and the groups of models that behave similarly can be seen in the critical difference diagram of figure 14. The p-values of the performed test used to construct the diagram are presented in table VIII. In this diagram it can be observed that the new models are clearly superior to the preexisting ones except for swARIMA, GMM and DeepAnT, but it is not clear which is the best model. This is probably because each model specializes in certain types of time series and certain types of anomalies.

V. Conclusion

All new models outperform the older pre existing models in absolute value but no statically. The new models may be slightly slower in most of the cases. This drawback is neglectable, as the improvement in F1-score performance is much higher. One of the techniques that seemed to be the most promising at first, DeepAnT, did not work that well in the end. Its architecture may be improved to predict better values. Taking into account the test's results, Matrix Profiling seems now one of the most promising techniques. But it needs to improve its time complexity issues. LRRDS probably can be improved if richer properties for each subsequence are extracted. For example PAPRRW's or EnrichedCCNAE's properties could be used. In fact, combining LRRDS subsequence extraction and PAPRRW's random walk model would probably return good results. However it may be slow or return bad results if the compression factor is too high. It seems also that researchers have focused mainly on univariate time series. Multivariate techniques underperform compared to univariate ones. It is clear that there is still a lot of research to be done in this area, as overall performance is still low. A model that is able to detect all types of anomalies does not exist and engineers have to relay on model ensembles. The number of researchers focusing on this topic is still low compared to other areas, but

TABLE V
F1-SCORE

	awa-1	awa-4	rac-0	rae-1	rkc-3	rkc-4	rtr-1	rtw-2	smd2-2-0	msl-0	mean
AAMP	77%	**81.25%**	**73.32%**	**58.82%**	-	32.35%	54.72%	**77.38%**	-	-	**64.98%**
ACAMP	**83.52%**	**81.25%**	22%	57.55%	-	40.31%	0%	-	-	-	47.44%
EnrichedCNNAE	5.78%	40%	50.45%	42.35%	12.14%	7.82%	65.36%	63.67%	16.23%	15.68%	31.95%
GMM	39.7%	-	4.31%	5.43%	48.17%	18.35%	2.4%	11.44%	-	-	18.54%
LRRDS	23.15%	0%	54.34%	13.63%	30.25%	23.27%	**65.06%**	25.31%	**21.17%**	**17.21%**	27.34%
PAPRRW	82.84%	38.17%	57.55%	50.08%	**64.25%**	**66.83%**	33.56%	38.98%	-	-	54.03%
swARIMA	21%	22.07%	47.28%	16.04%	0%	-	4.39%	37.1%	-	-	21.13%
DeepAnt	2.88%	19.21%	1.44%	5.75%	11.86%	8.66%	6.83%	4.38%	1.88%	6.06%	6.90%
AutoEncoder	2.45%	2.45%	4.54%	0%	4.64%	6.37%	0%	4.39%	1.58%	0.23%	2.67%
DAGM	4.8%	1.47%	0%	0%	1.63%	1.15%	0%	0%	1.41%	0.48%	1.09%
Isolation Forest	0.47%	0%	1.44%	6.66%	1.02%	0.38%	0%	1.59%	1.31%	0.38%	1.33%
Arima	0%	2.33%	0%	18.15%	0%	0%	0%	0.25%	-	-	2.59%
DeepPointAnomalyDetector	0%	1.82%	2.26%	0%	0%	0%	0%	5.92%	0%	0%	1.00%

TABLE VI
TRAIN + TRAIN PREDICTION IN SECONDS

	awa-1	awa-4	rac-0	rae-1	rkc-3	rkc-4	rtr-1	rtw-2	smd2-2-0	msl-0	mean
AAMP	0.27	0.38	**0.3**	0.17	-	7.39	0.07	14.12	-	-	452.84
ACAMP	0.49	0.48	0.32	0.32	-	25.27	0.06	-	-	-	903.37
EnrichedCNNAE	12.92	12.59	8.36	9.95	100.58	9.69	11.76	14.39	36.46	183.23	39.99
GMM	0.49	-	1.98	0.43	1.4	0.82	7	1.26	-	-	1.91
LRRDS	**0.19**	**0.19**	6.03	**0.07**	**0.1**	**0.27**	**0.03**	0.93	**0.96**	**5.87**	**1.46**
PAPRRW	3.5	5.35	5.03	2.95	24.72	9.44	2.01	18.62	-	-	8.95
swARIMA	1.36	4.46	6.89	5.93	8.53	-	15.64	30.043	-	-	10.41
DeepAnt	5.56	12.46	16.41	3.23	139.31	87.35	9.14	51.94	1166.43	4359.36	585.12
AutoEncoder	0.91	0.86	0.93	0.45	7.08	1.95	0.53	4.71	12.03	32.11	6.16
DAGM	2.59	1.64	1.76	0.83	7.67	3.81	1.43	8.52	54.56	128.37	21.12
Isolation Forest	0.33	0.28	0.32	0.39	0.61	0.73	0.37	**0.81**	4.18	10.55	1.86
Arima	0.48	0.94	1.16	0.27	2.17	2.64	3.236	1.23	-	-	1.52
DeepPointAnomalyDetector	7.24	1.45	1.49	0.78	7.4	4.06	0.97	5.8	53.72	144.51	22.74

it is growing year by year and soon enough performance of developed techniques will sky-rocket.

VI. PUBLISHED CODE

All model implementations in Merlion framework are available to use and contribute in https://github.com/alexfandos/Models4Merlion.

REFERENCES

[1] A. Blázquez-García, A. Conde, U. Mori, and Jose A. Lozano, "A review on outlier/anomaly detection in time series data," ACM Comput. Surv. 54, 3, Article 56, April 2022.

[2] D. M. Hawkins, "Identification of outliers," Springer Netherlands, 1980.

[3] R. Akbarinia, and B. Cloez, "Efficient Matrix Profile Computation Using Different Distance Functions, "CoRR abs/1901.05708, 2019.

[4] M. Munir, S. A. Siddiqui, A. Dengel, and S. Ahmed, "DeepAnT: A Deep Learning Approach for Unsupervised Anomaly Detection in Time Series,"IEEE Access 7, 2019.

[5] T. Kieu, B. Yang, and C. S. Jensen, "Outlier detection for multidimensional time series using deep neural networks," In Proceedings of the 19th IEEE, International Conference on Mobile Data Management, Aalborg, Denmark, 125–134, 2018.

[6] A. Reddy, M. Ordway-West, M. Lee, M. Dugan, J. Whitney, R. Kahana, B. Ford, J. Muedsam, A. Henslee, and M. Rao., "Using Gaussian Mixture Models to Detect Outliers in Seasonal Univariate Network Traffic," In 2017 IEEE Security and Privacy Workshops (SPW), IEEE, San Jose, CA, USA, 229–234, 2017.

[7] M. Hu, X. Feng, Z. Ji, K. Yan, and S. Zhou, "A novel computational approach for discord search with local recurrence rates in multivariate time series," Information Sciences 477, 220–233, 2019

[8] H. Ren, M. Liu, Z. Li, and W. Pedrycz, "A Piecewise Aggregate pattern representation approach for anomaly detection in time series," Knowledge-Based Systems 135, 29–39, 2017.

[9] Y. Zhou, R. Qin, H. Xu, S. Sadiq, and Y. Yu, "A Data Quality Control Method for Seafloor Observatories: The Application of Observed Time Series Data in the East China Sea," Sensors 18, 8, 2628, 2018

[10] A.v.d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "WaveNet: A Generative Model for Raw Audio", arXiv, 9, 2016.

[11] V. Ishimtsev, A. Bernstein, E. Burnaev, and I. Nazarov, "Conformal k-NN anomaly detector for univariate data streams," In Proceedings of the 6th Workshop on Conformal and Probabilistic Prediction and Applications, Vol. 60. PMLR, 213–227, 2017.

[12] J. Hochenbaum, O. S. Vallis, and A. Kejariwal, "Automatic anomaly detection in the cloud via statistical learning," 2017.

[13] Y. Zhu, C.C.M. Yeh, Z. Zimmerman, K. Kamgar,and E. Keogh, "Matrix profile xi: Scrimp++: Time series motif discovery at interactive speeds," In: 2018 IEEE International Conference on Data Mining (ICDM). pp. 837–846, 2018.

[14] J. Holešovský, M. Čampulová, and J. Michálek, "Semiparametric outlier detection in nonstationary times series: Case study for atmospheric pollution in Brno," Czech Republic. Atmosph. Pollut. Res. 9, 1, 27–36, 2018.

[15] H. Lu, Y. Liu, Z. Fei, and C. Guan, "An outlier detection algorithm based on cross-correlation analysis for time series dataset," IEEE Access 6 (2018), 53593–53610, 2018.

[16] K. Hundman, V. Constantinou, C. Laporte, I. Colwell, and T. Soderstrom, "Detecting spacecraft anomalies using LSTMs and nonparametric dynamic thresholding," In Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 387–395, 2018.

[17] P. M. Chau, B. M. Duc, and D. T. Anh, "Discord discovery in streaming

TABLE VII
TEST PREDICTION IN SECONDS

	awa-1	awa-4	rac-0	rae-1	rkc-3	rkc-4	rtr-1	rtw-2	smd2-2-0	msl-0	mean
AAMP	20.15	40.72	41.82	8.41	-	329.68	12.25	681.51	-	-	591.82
ACAMP	75.51	148.53	120.03	25.99	-	1166.87	39.15	-	-	-	1097.01
EnrichedCNNAE	**0.12**	0.22	**0.11**	**0.15**	1.21	**0.179**	0.16	**0.28**	0.58	5.1	0.81
GMM	0.63	-	1.31	0.31	1.9	1.04	5.91	1.32	-	-	1.77
LRRDS	0.31	0.6	27.65	0.94	0.54	1.83	0.2	14.26	0.83	5.74	5.29
PAPRRW	17.63	22.8	23.37	16.13	147.46	57.53	12.13	103.41	-	-	50.06
swARIMA	18.54	23.62	36.84	73.76	60.04	-	136.79	163.67	-	-	73.32
DeepAnt	1.57	1.55	1.74	0.44	39.48	8.6	0.32	2.26	63.44	325.19	44.46
AutoEncoder	0.808	0.52	1.23	0.08	7.72	0.51	**0.12**	4.3	**0.36**	**1.19**	1.68
DAGM	6.72	6.33	6.65	2.86	31.86	15.12	3.07	26.07	42.23	129.07	27.00
Isolation Forest	0.25	**0.18**	0.25	0.17	**0.88**	0.85	0.18	0.55	1.39	4.07	0.88
Arima	0.31	0.33	0.32	0.18	1.19	0.51	2.46	0.9	-	-	**0.78**
DeepPointAnomalyDetector	12.45	6.92	8.02	2.86	39.42	18.13	3.8	5.92	51.82	166.33	31.57

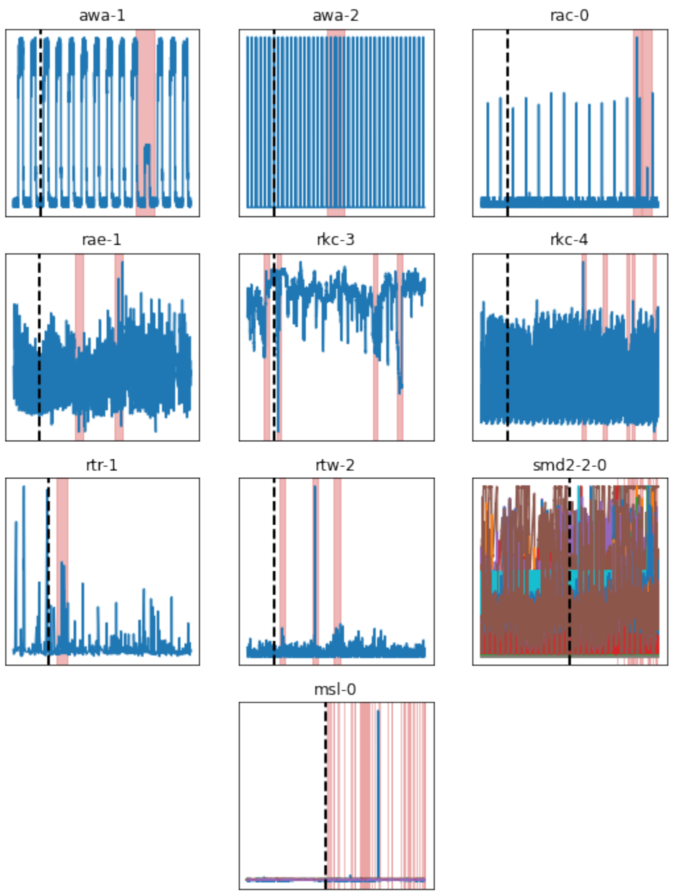

Fig. 12. The datasets that have been used to test the models. The dotted line points the train and test data separation.

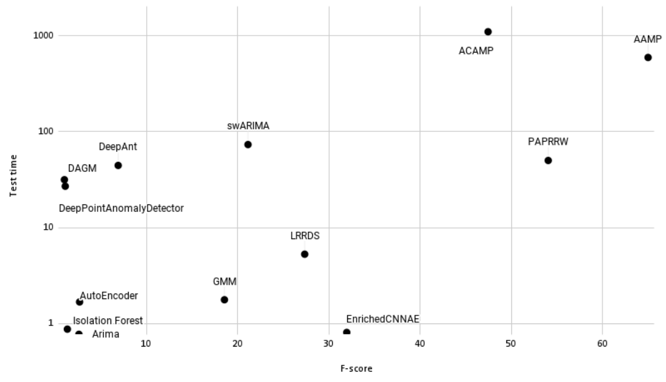

Fig. 13. F1-Score vs Test time

time series based on an improved HOT SAX algorithm," In Proceedings of the 9th International Symposium on Information and Communication Technology. ACM, 24–30, 2018.

[18] X. Wang, J. Lin, N. Patel, and M. Braun, "Exact variable-length anomaly detection algorithm for univariate and multivariate time series," Data Min. Knowl. Discov. 32, 6 (2018), 1806–1844, 2018.

[19] Y. Zhou, H. Zou, R. Arghandeh, W. Gu, and C. J. Spanos, "Non-parametric outliers detection in multiple time series a case study: Power grid data analysis," In Proceedings of the 32nd AAAI Conference on Artificial Intelligence. AAAI Press, 4605–4612, 2018.

[20] Y. Su, Y. Zhao, C. Niu, R. Liu, W. Sun, and D. Pei, "Robust anomaly detection for multivariate time series through stochastic recurrent neural network," In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'19). 2828–2837, 2019.

[21] S. E. Benkabou, K. Benabdeslem, and B. Canitia, "Unsupervised outlier detection for time series by entropy and dynamic time warping," Knowl. Info. Syst. 54, 2 , 463–486, 2018

[22] L. Beggel, B. X. Kausler, M. Schiegg, M. Pfeiffer, and B. Bischl, "Time series anomaly detection based on shapelet learning," Comput. Stat. 34, 3 (2019), 945–976, 2019

[23] J. A. Lara, D. Lizcano, V. Rampérez, and J. Soriano, "A method for outlier detection based on cluster analysis and visual expert criteria," Expert Syst. 37, 5 (2020), 1–23, 2020

[24] K. Shaukat, T. Mahboob Alam, S. Luo, S. Shabbir, I. A. Hameed, J. Li, S. K. Abbas, and U. Javed, "A Review of Time-Series Anomaly Detection Techniques: A Step to Future Perspectives," Advances in Information and Communication, FICC 2021, Advances inIntelligent Systems and Computing (Vol. 1363, pp. 865–877), Springer, 2021.

[25] A. Blázquez-García, A. Code, U. Mori, and J. A. Lozano, "A Review on Outlier/Anomaly Detection in Time Series Data", ACM Computing Surveys, Volume 54, Issue 3, Article No.: 56, pp1-33, 2022.

[26] M. Braei, and S. Wagner, "Anomaly Detection in Univariate Time-series: A Survey on the State-of-the-Art,", 2020

[27] H. Borges, R. Akbarinia, and F. Masseglia, ´´Anomaly Detection in Time Series," Transactions on Large-Scale Data- and Knowledge-Centered Systems, Springer Berlin / Heidelberg, In press., 2021.

[28] https://github.com/ibab/tensorflow-wavenet

[29] https://cran.r-project.org/package=otsad

[30] https://github.com/twitter/AnomalyDetection

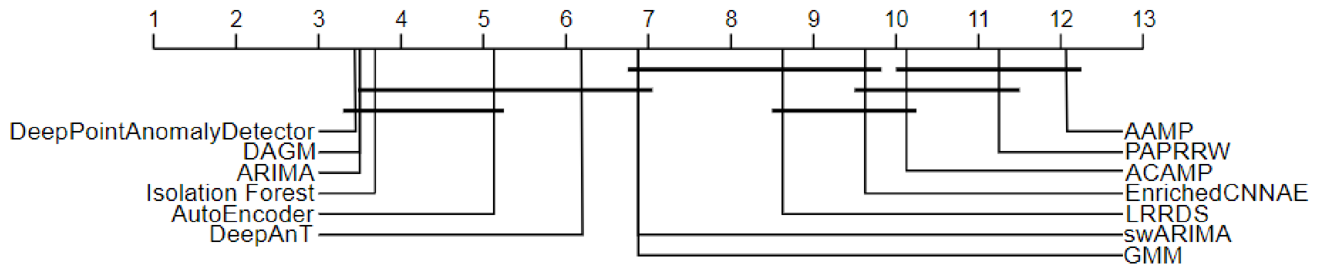

Fig. 14. Critical Difference Diagram

TABLE VIII
AFTER FRIEDMAN RANK SUM TEST POST-HOC PAIRED T-TESTS

Model1	Model2	p-value	Rejects H_0
AAMP	PAPRRW	0.255	No
AAMP	ACAMP	0.216	No
AAMP	EnrichedCNNAE	0.014	Yes
PAPRRW	ACAMP	0.692	No
PAPRRW	EnrichedCNNAE	0.243	No
PAPRRW	LRRDS	0.045	Yes
ACAMP	EnrichedCNNAE	0.416	No
ACAMP	LRRDS	0.198	No
ACAMP	GMM	0.015	Yes
EnrichedCNNAE	LRRDS	0.484	No
EnrichedCNNAE	GMM	0.206	No
EnrichedCNNAE	swARIMA	0.061	No
EnrichedCNNAE	DeepAnt	0.018	Yes
LRRDS	GMM	0.242	No
LRRDS	swARIMA	0.281	No
LRRDS	DeepAnt	0.041	Yes
GMM	swARIMA	0.834	No
GMM	DeepAnt	0.249	No
GMM	AutoEncoder	0.071	No
GMM	Isolation Forest	0.061	No
GMM	DAGM	0.041	Yes
swARIMA	DeepAnt	0.175	No
swARIMA	AutoEncoder	0.039	Yes
DeepAnt	AutoEncoder	0.076	No
DeepAnt	Isolation Forest	0.034	Yes
AutoEncoder	Isolation Forest	0.251	No
AutoEncoder	DAGM	0.079	No
AutoEncoder	Arima	0.859	No
AutoEncoder	DeepPointAnomalyDetector	0.087	No
Isolation Forest	DAGM	0.787	No
Isolation Forest	Arima	0.480	No
Isolation Forest	DeepPointAnomalyDetector	0.865	No
DAGM	Arima	0.572	No
DAGM	DeepPointAnomalyDetector	0.917	No
Arima	DeepPointAnomalyDetector	0.610	No

[31] https://cran.r-project.org/web/packages/envoutliers/index.html
[32] https://github.com/khundman/telemanom
[33] https://github.com/smallcowbaby/OmniAnomaly
[34] https://github.com/salesforce/Merlion
[35] https://facebookresearch.github.io/Kats/
[36] https://www.statista.com/statistics/871513/worldwide-data-created/
[37] https://www.3pillarglobal.com/insights/the-business-benefits-of-data-analytics-pose-enormous-opportunity/
[38] https://aimagazine.com/top10/10-ways-ai-can-be-used-smart-cities