Time Series Anomaly Detection

BOCHRA SMIDA, SALMA KASTALLI, and ZIED DAMMAK*, University of Passau, Germany

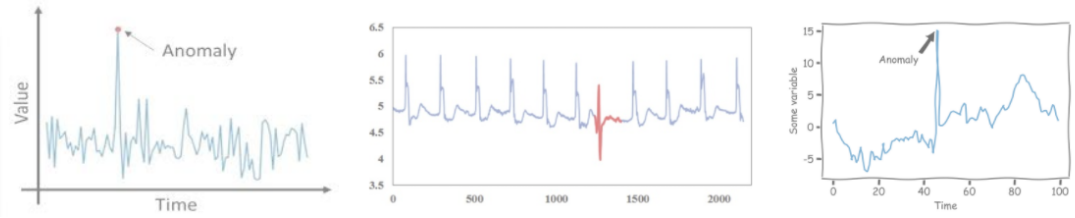

Fig. 1. Anomaly Detection for time series data.

Anomaly Detection is a technique used for identifying rare items, events, or observations that raise suspicions by differing significantly from the majority of the data. It is a challenging problem due to sparse occurrence of anomalous events, inconsistent behavior of different type of anomalies and imbalanced available data for normal and abnormal scenarios[?]. Numerous techniques have already been developed to mitigate this challenge. In this Artificial Intelligence Lab we want to evaluate the state-of-the-art methods that have already been developed in researches and came up with a new pipeline that treats 200 time series coming from different sources. The challenge we are presented with here is to come up with a method that solves the anomaly detection problem for univarirate multi-context datasets.

Additional Key Words and Phrases: time series ,anomaly detection, anomaly scores

ACM Reference Format:

Bochra Smida, Salma Kastalli, and Zied Dammak. 2022. Time Series Anomaly Detection. 1, 1 (February 2022), 11 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Many researchers have aimed to detect and analyze unusual, but interesting phenomena in time series data. The reason behind that is that abnormal or anomalous points can result in positive or negative impact on the project in hand. The challenge of this project is the need to build a model that should fit various types of time series with different characteristics. To tackle this problem we propose an exploratory data analysis phase that highlights the importance of smoothing the signal and that the choice of the window is important, after thank we will test our theory on different models. In this report we will start by the background section where we will define the terminologies and the techniques we will be using. Then in the state of the art we will evaluate some libraries that work on the same problem. In the methodology section, we will do first the Exploratory Data Analysis then test our theory on some models.

*All three authors contributed equally to this research.

Authors' address: Bochra Smida, bochra@ads.uni-passau.de; Salma Kastalli, salama01@ads.uni-passau.de; Zied Dammak, dammak01@ads.uni-passau.de, University of Passau, Passau, Bavaria, Germany, 933452.

1.1 Background

1.1.1 Time series. Time series are observations correlated in time. It is a series of data points indexed in time order. Most commonly, it is taken at successive equally spaced points in time.

1.1.2 Anomaly in time series data: Time series data can have outliers or anomalies. These are defines as "An observation which deviates so much from other observations as to arouse suspicions that it was generated by a different mechanism.". The detection of anomaly can be done by supervised learning algorithms if we have information on the anomalous behavior before modeling. But if initially we cannot identify these points, it is an unsupervised problem, which will be the topic of our paper. There are types of anomalies:

- Point anomalies – data point is too far from the rest, it falls into the category of point anomalies.
- Contextual anomalies – If the event is anomalous for specific circumstances (context)
- Collective anomalies- a collection of anomalous with respect to the whole dataset, but not individual objects.

1.2 The KDD Challenge

The goal of this competition is to encourage industry and academia to find a solution for univariate time-series anomaly detection. It has provided 200 data-sets collected over 20 years of research to further this area. We were also provided with a metadata file that contains the expected results that obey to these rules:

- Every time-series has exactly one anomaly.
- We can add +/- 100 locations on either side of the anomaly range to award the correct answer.

One type of model should fit various types of time series with different characteristics, such as trends and seasonalities. From the first challenge, the model parameters are assumed not to be adjusted for each time series. We can see from the figure 2, for the given time series record, we are required to find one specific abnormal point location.

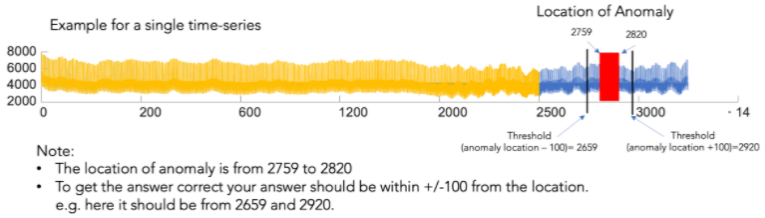

Fig. 2. Time Series Anomaly Detection dataset explanation (https://compete.hexagon-ml.com/practice/competition/39/#evaluation).

2 RELATED WORK

There are a lot of existing models. Our goal here is understanding how these models perform and to tune its parameters to improve the performance. In this section, we will go through all of the algorithms that we tested, we will define their main objective and test them on our data.

2.1 PyCaret

PyCaret's Anomaly Detection Module is an unsupervised machine learning module that is used for identifying rare items, events, or observations. It provides over 15 algorithms to analyze the results of trained models. The setup function

is the first and mandatory step to start the experiment in PyCaret. We tested all the models inside this library. Each iteration gave us a list of the anomaly scores, ranked from highest to lowest. For the evaluation, we looked at the top 30 highest anomaly scores and compared them with the results in the metadata file. For the phase one dataset 25 dataset some algorithms didn't perform well and others were time-consuming. So for the phase 2 dataset we selected these models that best worked with lower cost in the phase one:

- **Angle-base Outlier Detection**: It evaluates the degree of outlierness on the variance of the angles (VOA) between a point and all other pairs of points in the dataset.
- **Clustering-Based Local Outlier**: A measure for identifying the physical significance of an outlier.
- **Histogram-based Outlier Detection**: Assumes the feature independence and using histograms it calculates the degree of anomalies.
- **K-Nearest Neighbors Detector**: Proximity approach that classifies the new data or case based on a similarity measure.
- **Isolation forest**: It tries to separate each point by randomly creates a line in order to single out a point.

As shown in the table 1, we can now observe that for the original data, from the 200 datasets that the anomalies determined by the models selected give us a result of 76 out of 200. By this we mean that 76 out of 200 had the right anomaly point for the first 30 points detected. 41 point out of the 76 are correctly detected as the first anomaly that is what we call rank zero. The evaluation step for this method was unsuccessful, as we needed to come out with a method to find the best point of anomaly in our list of 30 top values. We will try to improve the result of this method in the methodology section.

2.2 Discord

Our research showed that the time series discords is one of the most effective and competitive methods in anomaly detection. It refers to the most unusual timeseries sub-sequences (maximally different). It uses a sliding window approach to find discords. It computes the distances for the windowed sub-sequence against the entire time series. A time series discord is a sub-sequence that is most dissimilar to its nearest neighbor or most significant discord. We can easily spot it from a Matrix Profile: the higher the Matrix Profile value, the greater the dissimilarity between the corresponding sub-sequence and its nearest neighbor, so the maximum value within a Matrix Profile indicates the time series discord. We tested this method on the phase one data. Each iteration gave us a list of the anomaly scores, ranked from highest to lowest. For the evaluation, we looked at the top 30 highest anomaly scores and compared them with the results in the metadata file. As shown in the table 1, we can now observe from the data from the second phase that the anomalies determined by this method selected gives us a result of 100 out of 200. The evaluation step for this method was also unsuccessful, as we needed to come out with a method to find the best point of anomaly in our list of 30 top values.

3 DATASET

After testing all the mentioned above methods, we want in this section to evaluate their results using two types of smoothing techniques.

3.1 Exploratory Data Analysis

Challenges in anomaly detection include defining normal behaviors, handling imbalanced distribution, addressing the variations in abnormal behavior and sparse occurrence of abnormal events. [**?**] The EDA step allows us to identify patterns and trend and investigate seasonality. We first want to test the distribution of all the data, then evaluate their stationarity and finally investigating the importance of the window size choice.

3.1.1 Box plots. :

Generating a boxplot quickly reveals, that some time series significantly dominates over the others. This proves that the dominance in magnitude of one data over the others should be taken into consideration in our algorithms. A solution for this is scaling so that the distribution become much more comparable as presented in the box plot of the figure 3.

Fig. 3. Rolling Mean & Standard Deviation

3.1.2 Stationarity of a Time Series. :

Stationarity is an important concept in time series analysis. It means that the statistical properties of a time series do not change over time. Many useful analytical tools and statistical tests and models rely on it. There are basic criterions to test it which are for example: the mean and the variance remaining constant over time. The figure 4 shows the seasonality trend of some of our time series. It is clear that our time series has seasonal variation, but we cannot perceive

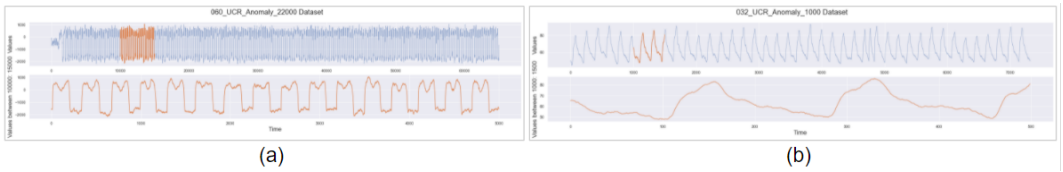

Fig. 4. Seasonality Analysis

the period. We have to test its stationarity using Rolling Statistics to calculate the rolling mean and variance using a specific window size.

The figure 5 shows us that the first criteria for stationary is not met as the mean is not constant (black line) . The second criteria of constant variance is also not met. As a result, we sure that our time series is not stationary.

Next we want to make time series stationary. We can use the Moving Average method, then the differentiating method.

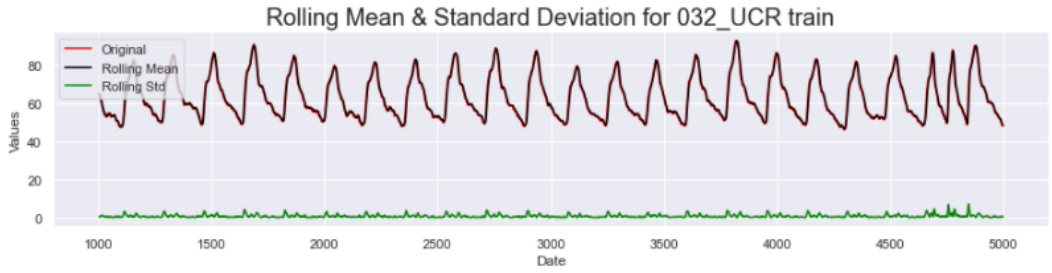

Fig. 5. Rolling Mean & Standard Deviation on the original data

Moving Average:

To solve the trend problem of the constant mean, one of the most popular methods is moving average, where we have a window that takes the average over the past 'w' sample. For this section, we vary the window size 'w' manually till we perceive a good result in the plot.

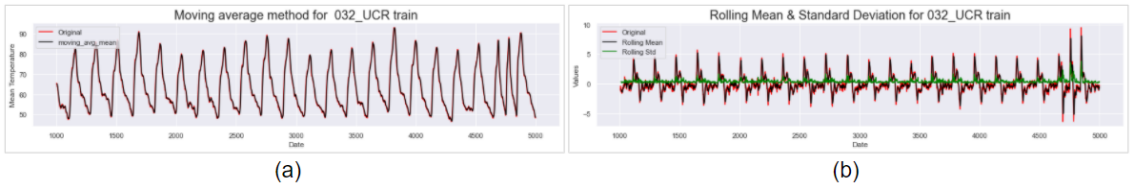

Fig. 6. (a) Moving Average on the data (b) Rolling Mean & Standard Deviation using moving average

The figure 6 presents our data after applying the Moving Average method. It shows that the first criteria for stationary is met. The mean looks like constant as you can see from plot (black line). Also the second criteria is met as the green line looks like constant. We achieve stationary time series using the Moving Average. Let's look now at another method to avoid trend and seasonality.

Differencing method: This method is about taking difference between time series and shifted time series. The figure 7 presents our data after applying the Moving Average method. It shows us that the first (constant mean) and second (constant variance) are met. So we can say with 99% confidence that this is a stationary series.

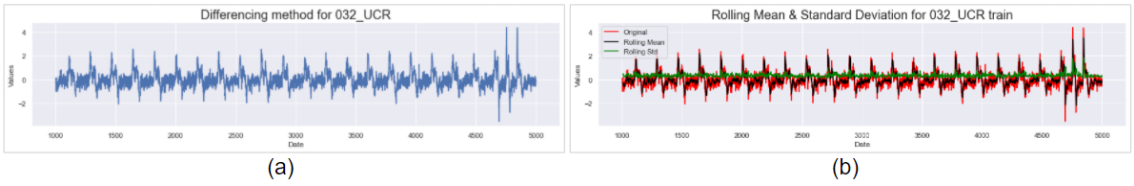

Fig. 7. (a) Using Differencing method on the data (b) Rolling Mean & Standard Deviation Using Differencing method

3.1.3 Analyzing Motifs. :

Time series motifs are approximately repeated sub-sequences found within a longer time series. The STUMPY library

can help us acquire motifs. STUMPY takes all sub-sequences within a time series can be compared by computing the pairwise z-normalized Euclidean distances. The index to the nearest neighbor is taken. This nearest neighbor distance vector is referred to as the matrix profile and the index to each nearest neighbor within the time series is referred to as the matrix profile index.

We are looking in the Figure 8 for the motif of the time series. We took 2 sections of the data and plotted them on each

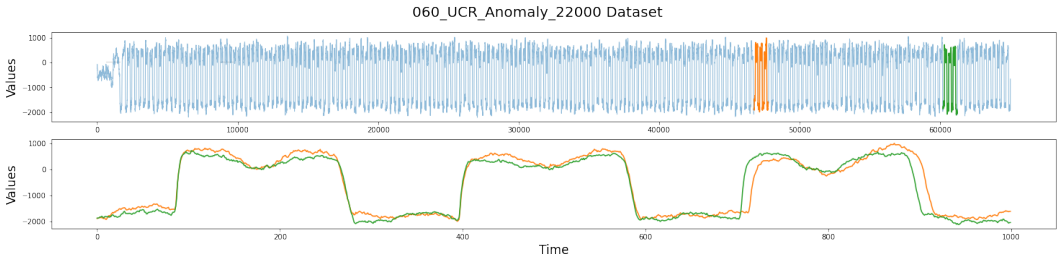

Fig. 8. Analyzing Motifs

other to try to see if they are a match. Doing this work manually, it is very hard to be certain that the orange and green sub-sequences are a match, that is, until we zoom in on them and overlay the sub-sequences on top of each other. This plot helped us visualize the data and figure out what window size we should use (length of one motif).

In the Figure 9, we used STUMPY to see what is the first detected motif and the nearest neighbor. We can clearly see

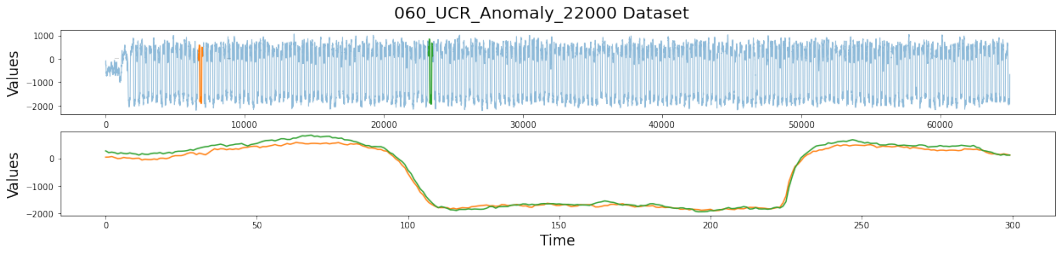

Fig. 9. Looking for the motif of the time series

that the motif is very similar. Computing the matrix profile allows you to quickly find motifs and its nearest neighbor. In here the choice of the Window was made visually.

To investigate the importance of the window size choice, we varied its size and tracked the change on the Matrix profile. We can see from Figure 10 that with varying window sizes, some peaks stay prominent and all the non-peak values are converging towards each other. This is why having a knowledge of the data-context is important prior to running stumpy. So it is helpful to have a window size that captures a repeating pattern or anomaly within the dataset.

Finding the right window size has always been a challenging task for many domains in data mining such as classification, clustering, motif discovery, anomaly detection, and time series prediction. The window size parameter is mostly given by experts or is based on domain knowledge. Failure to provide an appropriate window size may result in poor outcomes in underlying time series data mining tasks.

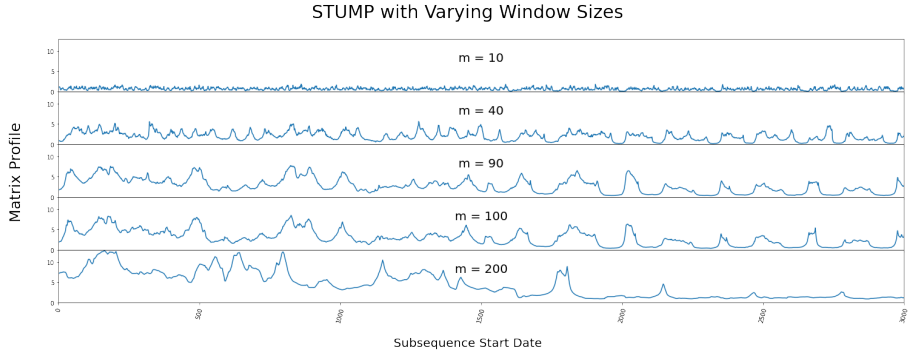

Fig. 10. Varying the size of the window and track the change on the Matrix profile

3.2 Model testing with different data entries

3.2.1 Using Confidence intervals. :

Using the confidence intervals for smoothed values we defined the max and min intervals to find the abnormal values outside these lines for the original data and then for the smoothed one from the last subsection. We used a simple anomaly detection system with the help of moving average to plot the difference for each dataset.

The Figure 11 shows that the work done in the EDA helped better the detection of the anomalies in time series. We can see that the anomalies (the red dots) in the sub figure (b) and (c) are better detected than in the original data.

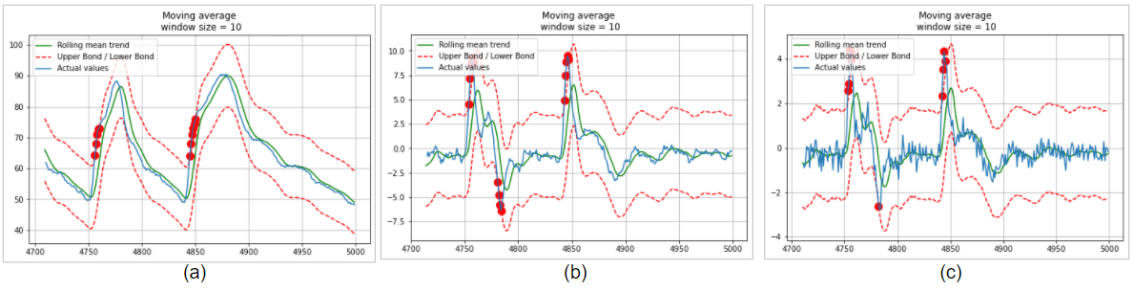

Fig. 11. Anomaly detection using Confidence intervals (a) with using original data, (b) using the Moving Average Method (c) using the Differentiating Method

3.2.2 Using PyCaret and Discord. :

The table 1 and 2 show the result of the anomaly detection using different data entries: the count of the data ids that were correctly detected and also the minimum rank of the position of the anomaly in the detected list. These results prove that the smoothing of the data done in the EDA section, can help better the detection of the anomalies in time series.

Table 1. PyCaret Results table

Data Method	*Top 20 Accuracy for phase 1 in Pycaret*	*Rank zeros for phase 1 in Pycaret*	*Top 20 Accuracy for phase 2 in Pycaret*	*Rank zeros for phase 2 in Pycaret*
Original data	10/25	2	76/200	41
moving avg diff	13/25	6	99/200	72
differencing method	14/25	8	115/200	74

Table 2. Discord Results table

Data Method	*Top 30 Accuracy for phase 1 in Discord*	*Rank zeros for phase 1 in Discord*	*Top 30 Accuracy for phase 2 in Discord*	*Rank zeros for phase 2 in Discord*
Original data	9/25	3	71/200	19
moving avg diff	13/25	6	71/200	20
differencing method	14/25	8	103/200	25

4 MAIN APPROACH: STATISTICAL METHOD FOR ANOMALY DETECTION

4.1 Anomaly scores

In this section we tried to solve the anomaly detection problem with a probabilistic approach. First, we define the main anomaly scores that we are going to use.

- **Amplitude:** In this approach we used the difference between the successive peaks in the original data and we calculate the difference between them while applying a rolling window w.
- **First order difference amplitude:** This metric represents the velocity of a signal. It calculates the rate of change of the position with respect to a frame of reference.
- **Second order difference amplitude:** This score indicates the rate of change of the velocity. We calculate this score for a window size w in order to find the points with high amplitude.
- **Interquartile scores:** This method finds out where the statistical significance is happening in our data. Thus we use the tukey test which helps in identifying the relationship between two parts of the signal. According to this method, a point is considered as an outlier if it is in greater than 1.5 of the interquartile range.
- **Z Score:** This score calculates the number of standard deviation that the desired point is greater or less than the mean of the distribution.
- **Standard deviation:** We calculate using a rolling window w the successive standard deviations in order to obtain the extreme values that will be considered as anomalous.

4.2 Architecture

The proposed pipeline presented in the figure 12 works as follow: The input data is used to generate multiple metrics that we have defined in the previous section. The anomalies are calculated using rolling window with multiple window sizes. Then, we ignore the insignificant scores using a threshold. The obtained scores are then smoothed using moving average. Finally, for each time series we apply an ensembling based on the rate of each anomaly metric. The chosen location of the anomaly is considered as the index of the maximum rate of the anomaly metric.

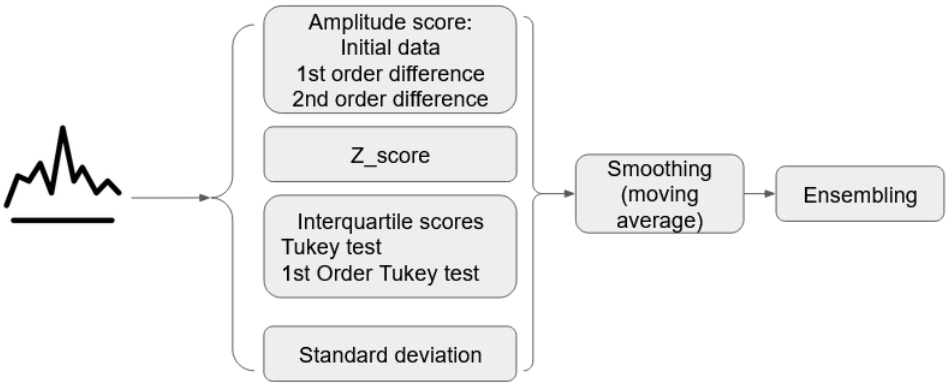

Fig. 12. Anomaly detection using probabilistic metrics

This approach helped us achieve 59 % of accuracy as mentioned in the table 3. According to the results we can verify the importance of the interquartile scores and amplitude which showed promising results in anomaly detection. The ZScore didn't perform well and this can be explained by the fact that it supposes that the data is normally distributed.

Table 3. Statistical anomaly detection

Method	***Interquartile***	***Amplitude***	***Zscore***	***Standard deviation***	***Total***	***Accuracy***
Value	32	55	1	29	117/200	59%

5 OTHER APPROACHES : MACHINE LEARNING APPROACHES

5.1 Autoencoder Method:

Because of their ability to train discriminative features automatically, deep neural networks (DNNs) are appealing alternatives to more traditional approaches for time series anomaly detection. An auto-encoder is a type of artificial neural network that learns how to code efficiently. An autoencoder's goal is to learn a compressed representation (encoding) for a set of data and extract important features as a result. In this method, we use autoencoder to predict data and compare the predicted data to our data to detect the anomaly. After testing the model using univariate time series, we got 6 out of 25.

To increase the performance of the model, we present a method for converting univariate time series into multivariate time series using a feature engineering strategy. In fact, studies prove that multivariate models produced a more precise and accurate prediction with reduced confidence intervals and improved accuracy metrics. As a result,it has been demonstrated that feature engineering improves anomaly detection models and that multivariate approaches on time series are more successful than univariate methods. [1]

Our technique involves extracting non-local information from the raw time series and constructing a new time series (i.e. another variate). To this end, we make use of the velocity, acceleration and matrix-Profile (MP) with multiple window sizes as different features to analyze more the time series.

The table 4 shows the accuracy of model using univariate data at first and then the accuracy of the model using our new technique with different non-local information.

Table 4. Autoencoder Results

Method	***Accuracy for phase 1 in autoencoder***	***Percent Accuracy***
Original data	6/25	24
Multi variate data	8/25	30

5.2 Local Outlier Factor Method:

LOF is a unsupervised machine learning algorithm based on the LRD: Local Reachability Density of the data points in the distribution to detect outliers. LOF compares the density of any given data point to the density of its neighbors. Anomalous point in the data will be located in low density zones and the LOF will be high. The accuracy of this model is presented in the table 5 and equals to 44 %.

Table 5. LOF Results

Method	Detected anomalies
LOF	87/200 (Accuracy: 44%)

6 EVALUATION

In this Lab, we want to develop a pipeline that analyses all the time series with their different characteristics and sources and finding one anomaly point. Our first approach was smoothing the data using Moving Average and the Differentiating Method. This step was proved to increase the accuracy of the Pycharet (from 20.5% to 36% to 39%) and Discord (from 9.5% to 10% to 12.5%) as it made the time series stationary. The disadvantage of the Pycharet and Discord method is in the evaluation phase. The Top 20 accuracy list shows a potential of 57.5% . The statistical methods on the other hand showed promising results as it used an ensembling approach. Combining different anomaly scores helped us achieve 59 % of accuracy. Other methods were also tested like the autoencoder with a max 30% accuracy and the Local Outlier Factor Method with a 44% accuracy. The Figure 13 displays or final results.

7 CONCLUSION AND FUTURE WORK

After the evaluation we noticed that some types of time series work better with certain anomaly scores. A supervised classification of the input data could help us decide what model to use for each class. This way the detection of the anomaly point would be done in less time. Also Predictive Deep learning approach are time-consuming due to lack of resources. without this limitation, adding more types of non-local information and using other types of predictive model such as LSTM autoencoders could predict better than the simple autoencoder we tested.

In this paper we presented different types methods to detect anomalies in a time series: Machine learning and statistical methods. We have shown that there are some ways to improve the performance of each model and we came to the conclusion that the best strategy is to combine the Statistical methods with the state of the art and the deep learning ones and develop a assembling method for evaluation.

REFERENCES

[1] Carta, S., Podda, A.S., Reforgiato Recupero, D.R., Saia, R. A local feature engineering strategy to improve network anomaly detection. Future Internet 12(10),177 (2020)

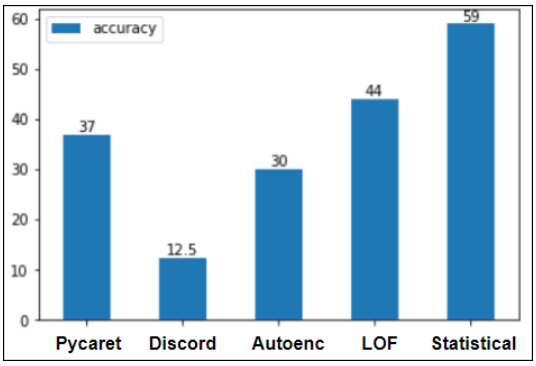

Fig. 13. Summery of the maximal accuracies reached in our pipeline

A TEAM CONTRIBUTION

A.1 Bochra

(1) State of the art model investigation.
(2) Model implementation and testing anomaly detection using Discord.
(3) Investigating the analysis using autoencoder:
- Univariate autoencoder model implementation.
- Investigating the multivariate analysis autoencoder and combining matrix profile and autoencoder for anomaly detection.

A.2 Salma

(1) Exploratory data analysis and Window evaluation :
- Investigating the stationarity of time series using the Moving average method and the differentiating method and test their influence of the anomaly detection when using methods like Confidence intervals, pycaret, discord and the statistical methods.
- Investigating the length of a motif using most prominent frequency with fft and Investigating the Impact of the window size using stumpy.

(2) Model implementation and testing using Confidence intervals and Pycaret

A.3 Zied

(1) Sate of the art models investigation
(2) Implementing and testing anomaly detection using statistical methods:
- Investigate the baseline statistical methods from the literature.
- Try various combinations for the ensembling of the found metrics.

(3) Model the problem as a proximity approach using LOF.