Machine Learning in Applications Time Series Anomaly Detection: SOTA Investigation

Ahmadreza Farmahini Farahani, Mona Davari, Omar Ormachea Hermoza
Politecnico di Torino

CONTENTS

List of Figures

List of Tables

Machine Learning in Applications Time Series Anomaly Detection: SOTA Investigation

***Abstract*—Despite the rapid advancement of classification algorithms, robots often produce anomalous behaviors that can lead to failures. The ability to detect such anomalous behaviors is a key component in modern robots to achieve high-levels of autonomy. Reactive anomaly detection methods identify anomalous task executions based on the current robot state and thus lack the ability to alert the robot before an actual failure occurs. Such an alert delay is undesirable due to the potential damage to both the robot and the surrounding objects.**

We review a set of anomaly detection approaches for time-series anomaly detection. First we tried Transformers which have shown great power in unified modelling of point-wise representation and pairwise association, and we find that the self-attention weight distribution of each time point can embody rich association with the whole series. Our key observation is that due to the rarity of anomalies, it is extremely difficult to build nontrivial associations from abnormal points to the whole series, thereby, the anomalies' associations shall mainly concentrate on their adjacent time points. This adjacent-concentration bias implies an association-based criterion inherently distinguishable between normal and abnormal points, which we highlight through the Association Discrepancy. Technically, we propose the Anomaly Transformer with a new adjustment mechanism to evaluate the data.

In the second step, we try two distinct outlier detection methods: COPOD (Copula Based Outlier Detection) and ECOD (Empirical Cumulative Distribution-based Outlier Detection). COPOD utilizes the concept of copulas, which are mathematical functions employed to model the joint distribution of multiple variables. By estimating the empirical copula and calculating tail probabilities, COPOD identifies outliers as data points exhibiting low tail probabilities, signifying extreme events deviating from the joint behaviour of the data. On the other hand, ECOD relies on empirical cumulative distribution functions (ECDFs) to estimate the cumulative distribution of a random variable. By calculating outlier scores based on the distances from the empirical cumulative distribution, ECOD detects outliers as data points with scores exceeding a predefined threshold, indicating significant deviations from the general distribution.

Finally, we try MTAD-GAT (Multivariate T.S. Anomaly Detection with Graph Attention Networks) which exploits the Graph Attention Networks architecture in order to explicitly model the correlation between features and their within-series temporal dependency. This is done by emulating a graph-like underlying structure of our time-series data.

I. Introduction

When it comes to analyzing data, outliers - which are data points with unique traits compared to the majority - can have a big impact on the results. That's why it's important to identify and manage these unusual data points in order to maintain the accuracy of data science models. Outlier detection algorithms are often used in real-world applications like detecting credit card fraud, network intrusion, and generating synthetic data. As a result, these algorithms need to have high detection accuracy, speedy execution, and be easy to interpret.

The field of supervised anomaly detection has seen significant advancements, but it is essential to also focus on unsupervised methods for predicting anomalies to prevent potential damages caused by robots.

This approach has the potential to yield economic benefits for companies. In real world systems, multiple measurements are generated and monitored by various sensors, making the detection of malfunctions from large scale monitoring data crucial for security and financial reasons. However, anomalies are rare and difficult to label, making unsupervised time series anomaly detection a challenging task. To overcome this challenge, the model must learn informative representations from complex temporal dynamics through unsupervised tasks and derive a criterion that can distinguish rare anomalies from the multitude of normal time points.

Numerous unsupervised OD algorithms have been proposed over the years These existing approaches have a few limitations.classic anomaly detection methods have provided many unsupervised paradigms, such as the density estimation methods and clustering based methods. These classic methods do not consider the temporal information and are difficult to generalize to unseen real scenarios.

Benefiting from the representation learning capability of neural networks, recent deep models have achieved superior performance. A major category of methods focus on learning pointwise representations through well-designed recurrent networks and are self-supervised by the reconstruction or autoregressive task. Here, a natural and practical anomaly criterion is the pointwise reconstruction or prediction error. However, due to the rarity of anomalies, the pointwise representation is less informative for complex temporal patterns and can be dominated by normal time points, making anomalies less distinguishable. Also, the reconstruction or prediction error is calculated point by point, which cannot provide a comprehensive description of the temporal context and may suffer from the curse of dimensionality both in detection accuracy and runtime efficiency and worsen rapidly as the number of data points and their dimensionality increase Second, most methods require hyperparameter tuning, which is difficult in the unsupervised setting.

This report will start with a literature review of the technologies, architectures and paradigms investigated during the development of this project. It will be followed by a section in which we will provide a formal introduction of the state-of-the-art methods implemented and tested, providing insights about their implementation. Then, a section will be dedicated

for reporting the experimental results obtained with the different models and discussing them.

II. Background

A. Unsupervised time series anomaly detection

As a vital real-world problem, unsupervised time series anomaly detection has been widely explored. Categorizing by anomaly determination criterion, the paradigms roughly include the density-estimation, clustering-based, reconstruction-based and autoregression-based methods.

1) Classic methods: Classic method, do not consider temporal information in time series. These methods (e.g. LOF, OC-SVM, SVDD) are hard to generalize to unseen data.

2) Proximity-based Algorithms: When it comes to detecting outliers, there are several algorithms that can be used. One popular approach is proximity-based algorithms, which rely on local neighborhood information around each point. These algorithms can be categorized into density- and distance-based algorithms. Density-based methods assume that an outlier can be found in a low-density region, whereas non-outliers are assumed to appear in dense neighborhoods. Distance-based methods, on the other hand, compare objects to their neighbors, and those that are considerably far from their neighbors are deemed outliers. While proximity-based methods are mostly nonparametric and intuitive, they can be computationally expensive and sensitive to hyperparameters.

3) Statistical Models: When it comes to outlier detection (OD), there are two main groups of statistical models: parametric and non-parametric methods. Parametric methods assume that the data come from a parametric distribution, while non-parametric methods do not assume any particular distribution. Parametric methods include Gaussian mixture models (GMM) and linear regression, while non-parametric methods include Kernel Density Estimation (KDE), histogram-based methods (HBOS), and others.

4) Recurrent networks: These networks are self-supervised by reconstruction and auto-regressive tasks. However, point-wise representation is less informative and can be dominated by normal points. Reconstruction or predictive error is point by point without comprehensive description.

5) Explicit association learning: These models are not sufficient for complex temporal patterns (e.g. vector auto-regression, state space models). They use sequence-based similarity calculation which cannot capture fine-grained association for each time point.

B. Transformers for time series analysis

Recently, Transformers (Vaswani et al., 2017) have shown great power in sequential data processing, such as natural language processing (Devlin et al., 2019; Brown et al., 2020), audio processing (Huang et al., 2019) and computer vision (Dosovitskiy et al., 2021; Liu et al., 2021). For time series analysis, benefiting from the advantage of the self-attention mechanism, Transformers are used to discover the reliable long-range temporal dependencies (Kitaev et al., 2020; Li et al., 2019b; Zhou et al., 2021; Wu et al., 2021). Especially for time series anomaly detection, GTA proposed by Chen et al. (2021) employs the graph structure to learn the relationship among multiple IoT sensors, as well as the Transformer for temporal modeling and the reconstruction criterion for anomaly detection. Unlike the previous usage of Transformers, Anomaly Transformer renovates the self-attention mechanism to the Anomaly-Attention based on the key observation of association discrepancy.

C. Anomaly transformer

For the time-series anomaly detection we use Anomaly Transformer [1]. We start analysing the main features of this network. We only change the evaluation method proposed in this paper.

D. Graph Attention Network

With the goal of efficiently dealing with various types of data, several attempts have been made in the literature for employing neural networks on graph-like structured data. The main limitation of these approaches is that they rely on the data having the same graph structure as the dataset with which the model was trained, making them non-generalizable. Graph Attention Networks (Velickovic et al. [2]) allow for node classification of graph-structured data exploiting the attention mechanism, which is the current standard for sequential data. The GAN architecture, through the *self-attention* mechanism, allows for parallel computation, applicability to graph nodes with different degree (number of edges), and is suitable for generalised, unseen graph structures. The novel component at the basis of this architecture is called *graph attention layer*.

E. Copula Models

Copulas are functions that enable us to separate marginal distributions from the dependency structure of a given multivariate distribution. Formally, a d-variate copula, $C : [0, 1]^d \rightarrow [0, 1]$, is the cumulative distribution function (CDF) of a random vector $(U_1, U_2, ..., U_d)$ with Uniform(0, 1) marginals:

$$C_U(u) = P(U_1 \leq u_1, ..., U_d \leq u_d) \tag{1}$$

It is well-known that uniform distributions can be transformed into any desired distributions via inverse sampling:

$$X_j = F_j^{-1}(U_j) \sim F_j \tag{2}$$

Sklar [**?**] shows that for random variables $(X_1, \ldots, X_d)$ with joint distribution function $F(x_1, \ldots, x_d)$ and marginal distributions $F_1, \ldots, F_d$, there exists a copula such that

$$F(\mathbf{x}) = C(F_1(x_1), \ldots, F_d(x_d)) \tag{3}$$

In other words, a copula allows us to describe the joint distribution of $(X_1, \ldots, X_d)$ using only their marginals. This gives a lot of flexibility when modelling high dimensional datasets, as we can model each dimension separately, and there is a guaranteed way to link the marginal distributions together to form the joint distribution. Sklar [**?**] also shows that if F has univariate marginal distri- butions $(F_1, \ldots, F_d)$, then there exists a copula C(.) such that equation 3 holds. Furthermore,

if the marginals are continuous, then C(.) can be uniquely determined. Similarly, by substituting the inverse of equation 2 into equation 1, we obtain the copula equation expressed in terms of the joint CDF and inverse CDFs.

$$\begin{aligned} C(u) &= P(FX_1(X_1) \leq u_1, ..., FX_d(X_d) \leq u_d) \\ &= P(X_1 \leq F_1^{-1}(X_1), ..., X_d \leq F_1^{-1}(X_d)) \\ &= F_X(F_1^{-1}(u_1), ..., F_1^{-1}(u_d)) \end{aligned} \tag{4}$$

Together, these two results are known as Sklar's [**?**] Theorem, and they guarantee the exists of a copula for any given multivariate CDF with continuous marginals, and provides a closed form equation for how the copula can be constructed.

F. ECDF

The term ECDF stands for Empirical Cumulative Distribution Function. In statistical analysis, an ECDF is a non-parametric estimator of the underlying cumulative distribution function of a random variable. It assigns a probability to each data point, ordering them from the smallest to the largest value.

The ECDF is calculated by plotting each data point in ascending order on the x-axis, with the y-axis representing the proportion of data points that have a value less than or equal to the corresponding x-axis value.

$$X = x_1, x_2, \ldots, x_n \; F_n(x) \qquad = \frac{1}{n} \sum_{i=1}^{n} 1x_i \leq x$$

Where X is a random sample of size n from a population with cumulative distribution function $F(x)$. $F_n(x)$ is the empirical cumulative distribution function. $1x_i \leq x$ is an indicator function that equals 1 if x_i is less than or equal to x and 0 otherwise. Properties of $F_n(x)$:

- $F_n(x)$ is a non-decreasing step function that takes values between 0 and 1.
- As $n \to \infty$, $F_n(x) \to F(x)$ for all x. The ECDF becomes an increasingly accurate estimate of the true CDF as the sample size increases.
- The jth order statistic $x_{(j)}$ in the sample corresponds to the point where $F_n(x)$ jumps by $\frac{1}{n}$.
- Percentiles and quartiles of the data can be estimated from $F_n(x)$.

G. Empirical Copula

COPOD uses a non parametric approach based on fitting empirical cumulative distribution functions (ECDFs) called Empirical Copula.

$$(\hat{U}1; i, ..., \hat{U}d; i) = (\hat{F}1(X1; i), ..., \hat{F}d(Xd; i) \tag{5}$$

Finally, by substituting the empirical copula observations into the first equality of equation 4

$$(\hat{C}(u_1, ..., u_d) = \frac{1}{n} \sum_{i=1}^{n} I(\hat{U}1; i \leq u_1, ..., \hat{U}d; i \leq u_d)) \tag{6}$$

Nelsen [**?**] show that an empirical copula, $\hat{C}(u)$ with multivariate CDFs supported on n points in the grid $\frac{1}{n}, \frac{2}{n}, \ldots, 1d$, has discrete uniform marginals on $\frac{1}{n}, \frac{2}{n}, \ldots, 1d$, and asymptotically converges to C(u) as a result of the central limit theorem.

H. evaluation

There has been different approaches proposed for evaluation. In original anomaly transformer paper (Jiehui et al., 2022), based on their approach, if a point in a contiguous anomalous segment is detected correctly, all anomalies in the same segment are also considered to have been correctly detected. This adjustment is justified by the observation that the time point causing the anomaly does not need to be exactly detected in practice. This approach also used in other papers (Shen et al., 2020).

III. Materials and methods

Anomalies are usually rare and hidden by vast normal time points, making labelling hard and expensive. In an unsupervised approach we need an informative model which provides a good representation for each time point. Moreover, it should have a distinguishable anomaly criterion. The general framework for anomaly detection consists on outputting a *anomaly score* for each time point in the time-series. The range of values of these scores is completely dependent on the underlying paradigm and architecture. Regardless of how this score is generated (could be based on forecasting, statistical or generative approaches), the criterion for predicting an anomaly is verifying if it satisfies the condition

$$score > threshold$$

where the $threshold$ is to be defined arbitrarily. It can be tuned with the help of a validation set or it can be estimated automatically with methods such as Peak-Over-Threshold [3].

Suppose monitoring a robot sensors of d sensors and recording the equally spaced observations over time. The observed time series X is denoted by a set of time points $x_1, x_2, ..., x_N$, where $x_t \in R^d$ represents the sensors data over time t. The unsupervised time series anomaly detection problem is to determine whether x_t is anomalous or not without labels. As aforementioned, we highlight the key to unsupervised time series anomaly detection as learning informative representations and finding distinguishable criterion. We use the Anomaly Transformer to discover more informative associations and tackle this problem by learning the Association Discrepancy, which is inherently normal-abnormal distinguishable.

In Fig. 1 we have time-series data captured by different sensors. Moreover, we have seasonal data that has specific repeating pattern through time.

A. Anomaly transformer

Given the limitation of Transformers (Vaswani et al., 2017 [4]) for anomaly detection, we use renovated vanilla architecture called Anomaly Transformer with an Anomaly-Attention mechanism. **Overall Architecture** Anomaly Transformer is characterized by stacking the Anomaly-Attention blocks and feed-forward layers alternately.

This stacking structure is conducive to learning underlying associations from deep multi-level features. Suppose the model

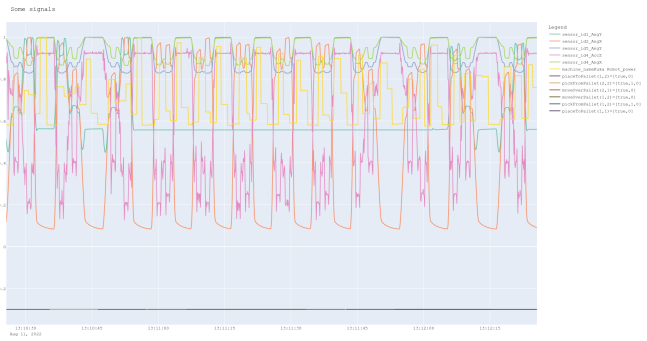


Fig. 1: Time-series data captured by different sensors

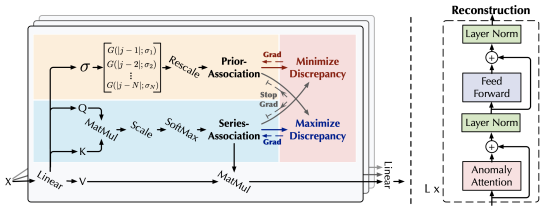


Fig. 2: Anomaly transformer architecture

contains L layers with length N input time series $X \in \mathbb{R}^{N \times d}$ The overall equations of the l-th layer are formalized as:

$$\begin{aligned} Z^l &= Layer - Norm(Anomaly - Attention(x^{l-1}) + x^{l-1})) \\ X^l &= Layer - Norm(Feed - Forward(Z^l) + Z^l)), \end{aligned} \tag{7}$$

where $x_l \in \mathbb{R}^{N \times d_{model}}$, $l \in 1, ..., L$ denotes the output of the l-th layer with d_{model} channels. The initial input $X^0 = Embedding(X)$ represents the embedded raw series. $z_l \in \mathbb{R}^{N \times d_{model}}$ is the l-th layer's hidden representation. Anomaly-Attention$(\cdot)$ is to compute the association discrepancy. As we can see in Fig. 2, in **Anomaly Attention** we have a two-branch structure. The initialisation is:

$$Q, K, V, \sigma = X^{l-1}W_Q^l, X^{l-1}W_K^l, X^{l-1}W_V^l, X^{l-1}W_\sigma^l \tag{8}$$

A temporal association is a distribution of association weights to all the time points along the temporal dimension. This method provide more informative for the temporal context, indicating temporal patterns, such as the period for trend of time series.

1) Prior-association: For the **prior-association**, a learnable Gaussian kernel have been adopted to calculate the prior with respect to the relative temporal distance. Benefiting from the unimodal property of the Gaussian kernel, this design can pay more attention to the adjacent horizon constitutionally. Moreover, a learnable scale parameter σ have been used for the Gaussian kernel, making the prior-associations adapt to the various time series patterns, such as different lengths of anomaly segments. Prior-association is formalized as follows:

$$P^l = Rescale\left(\left[\frac{1}{\sqrt{2\pi}\sigma_i} exp\left(-\frac{|j-i|^2}{2\sigma_i^2}\right)\right]_{i,j \in \{1,...,N\}}\right) \tag{9}$$

2) Series-association: The **series-association** branch is to learn the associations from raw series, which can find the most effective associations adaptively. Note that these two forms maintain the temporal dependencies of each time point, which are more informative than point-wise representation. They also reflect the adjacent-concentration prior and the learned real associations respectively, whose discrepancy shall be normal-abnormal distinguishable. The series-association of Anomaly-Attention in the l-th layer is:

$$S^l = Softmax\left(\frac{QK^T}{\sqrt{d_{model}}}\right) \tag{10}$$

And the reconstruction is as follows:

$$\hat{Z}^l = S^l V \tag{11}$$

Prior-association $P_l \in \mathbb{R}^{N \times N}$ is generated based on the learned scale $\sigma \in \mathbb{R}^{N1}$ and the i-th element σ_i corresponds to the i-th time point. Concretely, for the i-th time point, its association weight to the j-th point is calculate by the Gaussian kernel:

$$G(|j-i|; \sigma_i) = \frac{1}{\sqrt{2\pi}\sigma_i} exp\left(-\frac{|j-i|^2}{2\sigma_i^2}\right) \tag{12}$$

Further, we use $Rescale()$ to transform the association weights to discrete distributions P_l by dividing the row sum. $S_l \in \mathbb{R}^{N \times N}$ denotes the series-associations. In the Fig. 3 we have an example of how prior and series associations work.

Fig. 3: Temporal discrepancy

3) Association Discrepancy: We formalize the Association Discrepancy as the symmetrized KL divergence between prior- and series- associations, which represents the information gain between these two distributions (Neal, 2007). We average the association discrepancy from multiple layers to combine the associations from multi-level features into a more informative measure as:

$$AssDis(P, S; X) = \left[\frac{1}{L}\sum_{l=1}^{L}\left(KL(P_{i,:}^l || S_{i,:}^l) + KL(S_{i,:}^l || P_{i,:}^l)\right)\right]_{i=1,...,N} \tag{13}$$

From previous observation, anomalies will present smaller $AssDis(P, S; X)$ than normal time points, which makes $AssDis$ inherently distinguishable.

4) Minimax association learning: As an unsupervised task, we employ the reconstruction loss for optimizing our model. The reconstruction loss will guide the series-association to find the most informative associations. To further amplify the difference between normal and abnormal time points, we

also use an additional loss to enlarge the association discrepancy. Due to the unimodal property of the prior-association, the discrepancy loss will guide the series-association to pay more attention to the non-adjacent area, which makes the reconstruction of anomalies harder and makes anomalies more identifiable. The loss function for input series $X \in \mathbb{R}^{N \times d}$ is formalised as:

$$L_{Total}(\hat{X}, P, S, \lambda; X) = \|X - \hat{X}\|_F^2 - \lambda \times \|AssDis(P, S; X)\|_1 \tag{14}$$

When $\lambda > 0$, the optimization is to enlarge the association discrepancy. A minimax strategy is proposed to make the association discrepancy more distinguishable. At the minimize phase, the prior-association minimizes the Association Discrepancy within the distribution family derived by Gaussian kernel. At the maximize phase, the series-association maximizes the Association Discrepancy under the reconstruction loss.

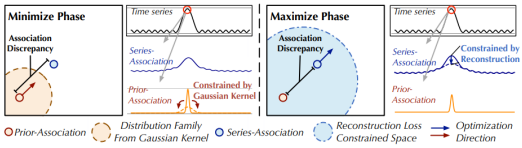


Fig. 4: Minimax optimization

Note that directly maximizing the association discrepancy will extremely reduce the scale parameter of the Gaussian kernel (Neal, 2007), making the prior-association meaningless. As we can see in Fig. **??** Towards a better control of association learning, a minimax strategy have been proposed. Concretely, for the **minimize phase**, we drive the prior-association P_l to approximate the series association S_l that is learned from raw series. This process will make the prior-association adapt to various temporal patterns. The loss function of this phase is:

$$L_{Total}(\hat{X}, P, S_{detach}, -\lambda; X) \tag{15}$$

For the **maximize phase**, we optimize the series-association to enlarge the association discrepancy. This process forces the series-association to pay more attention to the non-adjacent horizon. Thus, integrating the reconstruction loss, the loss function of this phases is:

$$L_{Total}(\hat{X}, P_{detach}, S, \lambda; X) \tag{16}$$

Under the reconstruction loss, this is much harder for anomalies to achieve than normal time points, thereby the normal-abnormal distinguishability of the association discrepancy have been amplified.

5) Association-based anomaly criterion: An incorporate of the normalized association discrepancy to the reconstruction criterion have been made, which will take the benefits of both temporal representation and the distinguishable association discrepancy. The final anomaly score of $X \in \mathbb{R}^{N \times d}$ is shown as follows:

$$\begin{aligned} AnomalyScore(X) =& Softmax\Big(-AssDis(P, S; X)\Big) \\ &\odot \Big[\|X_{i,:} - \hat{X}_{i,:}\|_2^2\Big]_{i=1,\ldots,N} \end{aligned} \tag{17}$$

where $\odot$ is the element-wise multiplication. $AnomalyScore(X) \in \mathbb{R}^{N \times 1}$ denotes the point-wise anomaly criterion of X . Towards a better reconstruction, anomalies usually decrease the association discrepancy, which will still derive a higher anomaly score. Thus, this design can make the reconstruction error and the association discrepancy collaborate to improve detection performance.

B. COPOD: Copula-Based Outlier Detection

COPOD is a novel outlier detection algorithm based on estimating tail probabilities using empirical copula.

COPOD is a three-stage process for outlier detection. $(F_1, \ldots, F_d)$ First, we compute the empirical CDFs based on the dataset of interest $X = [X_{1,i}, \ldots, X_{d,i}]$, $(i = 1, \ldots, n)$. Secondly, we use the empirical CDFs to produce the empirical copula function. Finally, we use the empirical copula to approximate the tail probability,

$$F_X(x_i) = P(X_1 x_{1,i}, ..., X_d x_{d,i}) \tag{18}$$

for each x_i .

1) Outliers as Tail Events: For every observation x_i , the goal is to compute the probability of observing a point at least as extreme as x_i . In other words, assume that x_i is distributed according to some d-variate distribution function F_X , we want to calculate $F_X(x_i) = P(X \leq x_i)$ and $1 - F_X(x_i) = P(X \leq x_i)$. If x_i is an outlier, it should occur infrequently, and as such, the probability of observing a point at least as extreme as x_i should be small. Therefore, if either $F_X(x_i)$ or $1F_X(x_I)$ is extremely small, there is evidence that this point has rare occurrence, and is, therefore, likely to be an outlier. We call $F_X(x_i)$ the left tail probability of x_i and $1F_X(x_i)$ the right tail probability of x_i . We say that an outlier has a small tail probability if either of the two quantities are small.

In order to compute the tail probabilities empirically, we first compute the empirical CDF , once we have the empirical CDFs, we can approximate the inverse of equation 2 to obtain the estimated copula observations u_j by feeding each x_j into F_j. The estimated copula observations tells us the probability of observing something as extreme as x along the j th dimension. Finally, multiplying all u_j 's together gives the empirical estimate of the left tail probability.

2) Computing Right Tail Probabilities: In a similar fashion, we can compute the right tail probabilities. All we needed is

$$C(1u) = P(U_1 u_1, ..., U_d u_d) \tag{19}$$

and since we are feeding $1u_j$ into $\hat{F}_j$, this should give us the desired result. However, because $\hat{F}_j$ is not a true cumulative distribution function, and it only has support from $\frac{1}{n}, \frac{2}{n}, \ldots, 1^d$. The largest u_j , in this case, would equal to 1, and substituting that into equation 19 implies that $P(U_1 \geq u_1, \ldots, U_j \geq$

$u_j, \ldots, U_d \geq u_d) = P(U_1 u_1, \ldots, U_j \geq 1, ..., U_d u_d) = 0$, as no uniforms can take on values greater than or equal to zero. A zero tail probability would make computing the outlier score difficult in subsequent steps. A simply way to address this challenge is to calculate $\hat{F}_j$ based on $X = [X_{1,i}, X_{2,i}, \ldots, X_{d,i}]$, and by calculating $\hat{F}_d$ (x). The desired result follows from substituting X_i and x . To avoid confusion in notations, we use $\hat{F}_j$ to denote the left tail ECDF derived from X and $\hat{F}_j$ to denote the right tail ECDF derived from X.

3) Diminishing Tail Probabilities: It is well known that various unusual phenomena arise in high-dimensional spaces, which do not occur in low-dimensional settings. This is known as the curse of dimensionality and outlier detection algorithms are known to be victims. Consider equation 6 in both low dimensional and high dimensional settings. As dimensionality increases, the probability of $\hat{U}_j, iu_j$, j decreases exponentially. Specifically,

$$\begin{aligned} \hat{C}(u1, \ldots, ud) &= \frac{1}{n} \sum_{i=1}^{n} I\{U_{1,i} \leq u1, \ldots, U_{d,i} \leq ud\} \\ &= \frac{1}{n} \sum_{i=1}^{n} I\{U_{1,i} \leq u1\} \times \cdots \times I\{U_{d,i} \leq ud\} \\ I\{U_{d,i} \leq ud\} &\approx P(U_{1,i} \leq u1) \times \cdots \times P(U_{d,i} \leq ud) \end{aligned} \tag{20}$$

and copula function approaches zero as d increases. In order to avoid diminishing tail probabilities, and using the monotonicity property of the log() function, we instead work with the sum of the negative log probabilities.

$$\begin{aligned} -\log(\hat{C}(u)) &= -\log(P(U_{1,i} \leq u_1) \times \cdots \times P(U_{d,i} \leq u_d)) \\ &= -\sum_{j=1}^{d} \log(P(U_{j,i} \leq u_j)) = -\sum_{j=1}^{d} \log(u_j) \end{aligned} \tag{21}$$

The last line holds since Ũ1,i is uniformly distributed on[0, 1].Notice that the motivation of taking logs is similar to that of the log likelihood functions. We point out a need for correcting for skewness, in cases where outliers fall on one extreme end of the distribution. we note that the skewness of the dataset plays a major role in whether left tail ECDFs or right tail ECDFs should be used. there for for each dimention we calculate the skewness vector as $b = [b_1, \ldots, b_d]$ where

$$(b_i) = \frac{E[(X - E[X])^3]}{(E[(X - E[X])^2])^{3/2}} \tag{22}$$

4) Advantages of COPOD: This is the list of the main advantages of using COPOD algorithm

- is deterministic without hyper-parameters and avoids challenges in hyper-parameters selection.
- highly interpretable through the Dimensional Outlie Graph, providing guidance on the causes of abnormality.
- is efficient and scales well for high-dimensional datasets. in are case of study it helps especially in the frequency 100 and 200 that the amount of data increases significantly.

Algorithm 1: Copula Outlier Detector

Require: input data X
Ensure: outlier scores $O(X)$
0: **for all** each dimension d **do** Compute left tail ECDFs: $\hat{F}d(x) = \frac{1}{n} \sum i = 1^n I(X_i \leq x)$ Compute right tail ECDFs: $\hat{\bar{F}}d(x) = \frac{1}{n} \sum i = 1^n I(-X_i \leq -x)$ Compute the skewness coefficient according to Equation 22.
0: **end for**
for $i = 1, \ldots, n$ **do**
Compute empirical copula observations
$\hat{U}_{d,i} = \hat{F}_d(x_i)$,
$\hat{V}_{d,i} = \hat{\bar{F}}_d(x_i)$,
$\hat{W}_{d,i} = \hat{U}_{d,i}$ if $b_d < 0$ otherwise $\hat{V}_{d,i}$
Calculate tail probabilities of X_i as follows:
$p_l = -\sum_{j=1}^{d} \log(\hat{U}_{j,i})$,
$p_r = -\sum_{j=1}^{d} \log(\hat{V}_{j,i})$,
$p_s = -\sum_{j=1}^{d} \log(\hat{W}_{j,i})$,
Outlier Score $O(x_i) = \max\{p_l, p_r, p_s\}$,
end
0: Return $O(X) = [O(x_1), \ldots, O(x_d)]^T$ =0

Algorithm 2: ECOD algorithm

Require: input data $X = \{X_i\}_{i=1}^n \in \mathrm{R}^{n \times d}$ with n samples and d features; $X_i^{(j)}$ refers to the value of j -th feature of the i -th sample
Ensure: outlier scores $O := \mathrm{ECOD}(X) \in \mathrm{R}^n$
0: **for** each dimension j in 1,...,d **do** Estimate left and right tail ECDFs (using equations (1) and 2, which we reproduce below): ,

left tail ECDF:$F_{\text{left}}^{(j)}(z) = \frac{1}{n} \sum_{i=1}^{n} 1\{X_i^{(j)} \leq z\}$ for $z \in R$

,

right tail ECDF:$F_{\text{right}}^{(j)}(z) = \frac{1}{n} \sum_{i=1}^{n} 1\{X_i^{(j)} \geq z\}$ for $z \in R$

0: **end for**
0: **for** each sample i in 1, . . . , n **do** Aggregate tail probabilities of X_i to obtain outlier score O_i: ,
$O_{\text{left-only}}(X_i) = -\sum_{j=1}^{d} \log(bF_{\text{left}}^{j}(X_i^j))$,
$O_{\text{right-only}}(X_i) = -\sum_{j=1}^{d} \log(bF_{\text{right}}^{j}(X_i^j))$, $O_{\text{auto}}(X_i) = -\sum_{j=1}^{d} \left[1_{\{\gamma_j < 0\}} \log(bF_{\text{left}}^{j}(X_i^j)) + 1_{\{\gamma_j \geq 0\}} (bF_{\text{right}}^{j}(X_i^j))\right]$
,
Set the final outlier score for point X_i to be
$O_i = \max\{O_{\text{left-only}}(X_i), O_{\text{right-only}}(X_i), O_{\text{auto}}(X_i)\}$
0: **end for**
1: **return** $O(X) = [O(X_1), \ldots, O(X_n)]$ =0

C. ECOD: Empirical Cumulative Distribution Functions

The ECOD (Empirical-Cumulative-distribution-based Outlier Detection) is an innovative unsupervised outlier detection algorithm that excels in identifying data points that deviate from a general data distribution.

Drawing inspiration from the concept that outliers are usually the "rare events" that appear in the distribution tails, ECOD estimates the underlying distribution of input data initially.

This method does not require prior selection of any parameters and approximates the entire data distribution without making any parametric assumptions. However, a challenge arises when working with high-dimensional data due to the slower convergence of the joint ECDF over all variables to the true joint CDF as the number of dimensions increases. To address this, ECOD computes a univariate ECDF for each dimension separately. The outlines of a data point are then measured by computing its tail probability across all dimensions through an independence assumption. This involves multiplying all the estimated tail probabilities from the different univariate ECDFs.

ECOD proceeds through two main steps. First, we calculate each dimension's left- and right-tail ECDFs. Next, for every point X_i, we aggregate its tail probabilities $bF(j)_{\text{left}}(X(j)_i)$ and $bF(j)_{\text{right}}(X(j)_i)$ to derive a final outlier score $O_i \in [0, \infty)$; higher means more likely to be an outlier. Note that these outlier scores are not probabilities, but are meant to be used for comparison between different data points.

1) Advantages of ECOD: The ECOD method used for Outlier Detection (OD) comes with several advantages:

- ECOD estimates the empirical cumulative distribution function (ECDF) of the data, which does not require any parameters to be tuned and approximates the entire distribution without making any parametric assumptions.
- Despite the technical difficulty in using ECDFs with high-dimensional data, ECOD sidesteps this issue. It computes a univariate ECDF for each dimension separately.
- For a data point, its outlyingness is measured by computing its tail probability across all dimensions. This involves multiplying all the estimated tail probabilities from the different univariate ECDFs.
- ECOD accommodates the skewness of the dataset for selecting tail probabilities. For dimensions that skew positively, right tail probabilities are used, whereas for dimensions that skew negatively, left tail probabilities are chosen.
- A skewness corrected (SC) version of ECOD is implemented for tail selection based on each dimension's skewness, resulting in accurate outlier detection.

D. MTAD-GAT

Multivariate Time-series Anomaly Detection via Graph Attention Network (MTAD-GAT) (Zhao et al., 2020 [5]) is a framework that aims to model multivariate correlations between different time-series explicitly. The idea behind is to consider every feature of the multivariate time-series as an individual time-series and try to model the correlations between different features explicitly, while independently modelling the temporal dependencies within each time-series.

MTAD-GAT's architecture is based on two graph attention layers [2]. The *feature-oriented graph attention layer* captures the causal relationship between features and the *time-oriented graph attention layer* captures the within-series temporal dependency. The training is then carried over a *forecasting-based model* and a *reconstruction-based model* simultaneously, through a suitable joint objective function. The overall structure is represented in Figure 5.

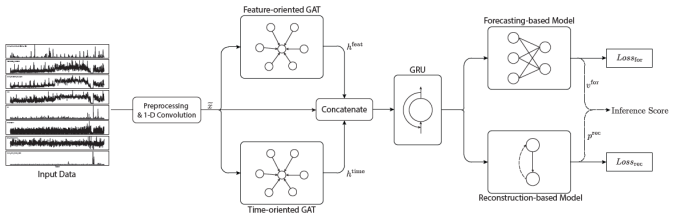


Fig. 5: MTAD-GAT architecture

1) Graph Attention Layer: A GAT layer models the relationship between nodes in arbitrary graphs. Generally, given a graph with n nodes, each node is represented by $\{v_1, v_2, ..., v_n\}$, where v_n is the feature vector of each node, a GAT layer computes the output representation for each node as follows:

$$h_i = \sigma(\sum_{j=1}^{L} \alpha_{ij} v_j)$$

where h_i denotes the output representation of the node i, which has the same shape with input v_i; σ represents the sigmoid activation function; α_{ij} is the attention score which measures the contribution of node j to node i, where j is one of the adjacent nodes for node i; L denotes the number of adjacent nodes for node i. The attention score α_{ij} can be computed by the following equations:

$$e_{ij} = \text{LeakyReLU}(w^T \cdot (v_i \oplus v_j))$$

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{l=1}^{L} \exp(e_{il})}$$

In the above equation, $\oplus$ represents concatenation of two node representations, $w \in R^{2m}$ is a column vector of trainable parameters, m is the feature vector dimension of each node, and LeakyReLU is a nonlinear activation function.

Now, we introduce the two main graph attention layers of MTAD-GAT:

a) Feature-oriented graph attention layer: The full multivariate time-series is treated as a fully-connected (complete) graph. Each node represents a given feature, while each edge represents the relationship between the nodes that it connects. The graph structure is reported in Figure 6. Each node x_i is represented by a sequential vector $x_i = \{x_{i,t} | t \in [0, n)\}$ and there are k nodes in total, where n is the $total$ number of timestamps and k is the total number of multivariate features.

b) Time-oriented graph attention layer: On the other hand, this layer considers all timestamps within a sliding window of arbitrary size as a complete graph. A node x_t represents the full feature vector at timestamp t, while its adjacent nodes represent all the remaining timestamps within the sliding window. This resembles the Transformer's [4] self-attention mechanism, were the complete word sequence is fully connected through it.

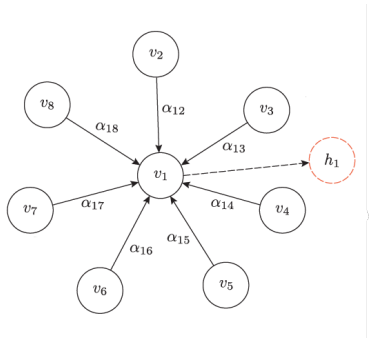


Fig. 6: Feature-oriented graph attention layer. Dashed circle is the final output.

The *feature-oriented graph attention layer* outputs a $k \times n$ matrix, whereas on the *time-oriented graph attention layer* the output has a shape of $n \times k$. The two outputs are concatenated together with the input time-series (after applying some pre-processing) which yields a matrix with shape $n \times 3k$, where each row represents a $3k$ dimensional feature vector for each timestamp, capturing fused information from different sources.

2) Joint optimized models: As aforementioned, the optimization objective concerns two separate models, one *forecasting-based* and the other being *reconstruction-based*. The overall loss function is expressed simply as:

$$Loss = Loss_{for} + Loss_{rec}$$

where $Loss_{for}$ denotes the loss function of the forecasting-based model and $Loss_{rec}$ denotes the loss function of the reconstruction-based model.

a) Forecasting-based model: This model predicts the value at the next timestamp. It consists of a stacking of three fully-connected layers after a GRU (Gated Recurrent Unit) layer. Its corresponding loss function can be formulated as a Root Mean Square Error between predicted and actual values:

$$Loss_{for} = \sqrt{\sum_{i=1}^{k}(x_{n,i} - \hat{x}_{n,i})^2}$$

Where x_n denotes the next timestamp for the current input $x = (x_0, x_1, ..., x_{n-1})$; $x_{n,i}$ represents the value for the i^{th} feature in x_n and $\hat{x}_{n,i}$ is the predicted value.

E. Anomaly transformer

a) Reconstruction-based model: The authors employ a Variational Auto Encoder (VAE) in the original paper, which provides a probabilistic manner for describing an observation in a latent space. Treating the values of time-series at each timestamp as variables, the VAE model is able to learn its entire data distribution. That is, given an input x (complete time-series), it should be possible to reconstruct it from a conditional distribution

$$p_\theta(z|x) = \frac{p_\theta(x|z)p_\theta(z)}{p_\theta(x)}$$

where the marginal probability density is expressed as

$$p_\theta(x) = \int p_\theta(z)p_\theta(x|z)dz$$

In order to calculate the above equation, a recognition model $q_\phi(z|x)$ is introduced for approximating the posterior distribution. Given said recognition model (encoder) $q_\phi(z|x)$ and the generative model $p_\theta(\hat{x}|z)$ (decoder), the loss function can be expressed as follows:

$$Loss_{rec} = -\mathrm{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] + D_{KL}(q_\phi(z|x)||p_\theta(z))$$

where the first term is the negative of the expected log-likelihood of the given input. The second term in the equation is the Kullback-Leibler divergence between the recognition model distribution $q_\phi(z|x)$ and $p_\theta(z)$, which acts as a regularisation term.

1) Scoring and model inference: Optimising the two aforementioned models implies that each of them will provide an output at the same time. The forecasting value will output a predicted value, while the reconstruction model will output reconstruction probabilities. The authors propose an inference score s_i for each feature and take the summation of all features as the final inference score. A given timestamp is predicted as anomalous if its corresponding inference score is larger than a threshold. This score can be calculated by:

$$score = \sum_{i=1}^{k} s_i = \sum_{i=1}^{k} \frac{(\hat{x}_i - x_i)^2 + \gamma(1 - p_i)}{1 + \gamma}$$

where $(\hat{x}_i - x_i)^2$ is the squared error between the forecasted value $\hat{x}_i$ and the actual value x_i; $(1 - p_i)$ is the probability of encountering an abnormal value for the i^{th} feature according to the reconstruction model; k is the total number of features; γ is a hyper-parameter to combine the two outputs.

2) Implementation: For testing the model with our data (Kuka dataset) we introduce some configurations used. For preprocessing the data, we apply Z-score normalization and Variance Threshold Feature Selection in order to remove all features whose variance doesn't meet some threshold. By default, it removes all zero-variance features, i.e., features with the same value in all samples.

F. Evaluation approachs

1) First method: In the MTAD-GAT paper [5], the evaluation used follows an approach described by Ren et al. (2019) [6]: The whole segment of continuous anomalies is treated as a positive sample, only one effective detection will be counted. If any point in an anomaly segment can be detected by the algorithm, and the delay of this point is no more than k from the start point of the anomaly segment, the full segment is assumed to have been detected correctly. Thus, all points in this segment are treated as True Positives, and the points outside the anomaly segments are treated normally.

2) Proposed evaluation method: In the original anomaly transformer paper (Jiehui et al., 2022) they have used point-wise anomaly detection as described before. We propose a new approach for evaluation. In the original paper, they have used a specific window size with a stride size equal to that window size. Since the anomalies of KUKA robot are contextual we used window sizes with $stride = 1$.

Let's assume we have an input data sequence $x_1, x_2, ..., x_N$, where $x_t \in \mathbb{R}^d$. We split the data with window size W_s.

Algorithm 3: Evaluation algorithm

Data: $\tilde{A_(\tilde{x_t})}$ where $\acute{N} \geq 0$, $th \in \mathbb{R}$, $gt \in \mathbb{R}^N$, $W_s \in \mathbb{N}$
Result: confusion matrix
$n \leftarrow 0$;
while $\tilde{A_(\tilde{x_t})}$ *is not None where* $t = n$ **do**
  if 1 *in* $gt[n : n + W_s]$ **then**
    /* We need to find end of the contextual anomaly */
    $c \leftarrow 0$;
    while $gt[n + W_s + c] = 1$ **do**
      $c \leftarrow c + 1$;
    end
    if $t \in [n : n + W_s + c] : \tilde{A_(\tilde{x_t})} \geq th$ **then**
      $detected \leftarrow true$;
    else
      $detected \leftarrow false$;
    end
    if $detected = true$ **then**
      $win\_count \leftarrow 0$;
      while $n + win\_count < n + W_s + c$ **do**
        if $gt[n + win\_count] = 1$ **then**
          $prediction \leftarrow True\ Positive$;
        end
        $win\_count \leftarrow win\_count + 1$;
      end
    else
      $win\_count \leftarrow 0$;
      while $n + win\_count < n + W_s + c$ **do**
        if $gt[n + win\_count] = 1$ **then**
          $prediction \leftarrow False\ Negative$;
        else
          $prediction \leftarrow True\ Negative$;
        end
        $win\_count \leftarrow win\_count + 1$;
      end
    end
    $n \leftarrow n + win\_count + c$;
  else
    if $\tilde{A_(\tilde{x_t})} \geq th$ *where* $t = n$ **then**
      $prediction \leftarrow False\ Positive$;
    else
      $prediction \leftarrow True\ Negative$;
    end
    $n \leftarrow n + 1$;
  end
end

Windows have a stride size S. The new transformed data is $\tilde{x_t}$, where it can be denoted as $\tilde{x_1}, \tilde{x_2}, ..., \tilde{x^{`}}\ N$. Furthermore, we have $\tilde{x_t} \in \mathbb{R}^{W_s \times \tilde{d}}$ and $\tilde{d} = d - W_s$. The length of the data $\acute{N}$ is constrained by the S and W_s and it can be formulated as follows:

$$\acute{N} = \lfloor \frac{N - W_s}{S} \rfloor \tag{23}$$

For calculating total anomaly score for a specific window $\tilde{A_(\tilde{x_t})}$, we first calculate each data points anomaly scores separately $A_(x_t)$. Then for each window we calculate summation of all the data points. Therefor, we have a summed anomaly score for each $\tilde{x_t}$. The calculations are as follows:

$$\begin{aligned} A_(x) &= AnomalyScore(x) \\ \tilde{x_t} &= [x_1, ..., x_i], i = W_s \\ \tilde{A_(\tilde{x_t})} &= \sum_{i=1}^{W_s} AnomalyScore(x_i) \forall x_i \in \tilde{x_t} \end{aligned} \tag{24}$$

After calculation of each windows anomaly scores, we need an optimised algorithm to analyse our model performance. We have a threshold th for detecting anomalies. If the $\tilde{A_(\tilde{x_t})}$ is above this certain threshold we have anomalies in the specified window. Based on the given ground truth gt we calculate the confusion matrix.

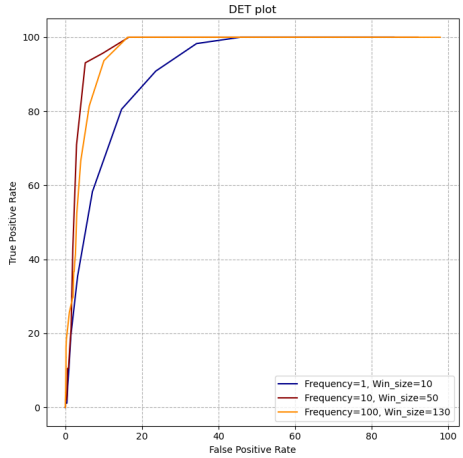
IV. Results

In this section we will report the results obtained by training and testing over the presented models. The main difficulties encountered concerned the high frequency sampled time-series, 100 Hz and 200 Hz. These time-series contained an immense amount of data points, and naturally required to increase the windowing size proportionally, in order to capture similar time intervals than those analysed with the lower frequency recordings. It should be mentioned that during the analysis of the data we use fBeta-score as the main comparison factor as in the context of are application recall is the main subject of are analysis .therefore, we chose the impotence of recall as twice of Precision . We start analysing the results using proposed methods.

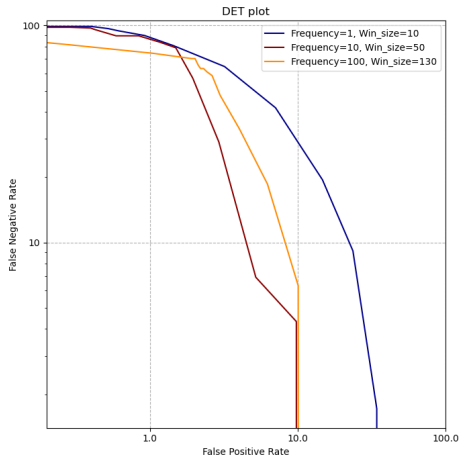
A. Anomaly Transformer

In the case of anomaly detection, since the not anomaly data is significantly higher than anomaly data, we expect low precisions. However, we should be careful that this metric do not exceed a certain threshold. In a real world application, whenever an anomaly is being detected, a worker can check whether the robot working correctly or not. We should consider recall and speccificity the most important factors.

We start analyzing anomaly transformer by using different values of window size and anomaly ratio for different frequencies. As we can see in Table. **??**, for $frequency = 1$ we do not have results. In this frequency associations cannot



(a) ROC curve



(b) DET plot

Fig. 7: ROC curve and DET plot for anomaly transformer.

be calculated due correctly due to loss of information in each timestamp.

In $frequency = 10$ we obtain best performance within $Win\_size = 50$. However, we have good results for different window sizes. Choosing window size is not critical for the application.

In $frequency = 100$ we have the best performance within $Win\_size = 130$. It is not appropriate to compare models with different frequencies.

In Fig. 7 we have ROC curve and DET plot for the three frequencies with their corresponding window sizes. However, since we have an unsupervised application, we have to compute the threshold by specifying the anomaly ration. Anomaly ratio is the expected anomaly that we can have based on train and test data without considering labels.

For future applications, we can use the ROC curve and DET plot to choose the best threshold effectively. However, if we do not have access to labels, we can select the threshold by anomaly ratios as we saw in Table. **??**.

Win_size	$Accuracy$	$Recall$	$Specificity$	$Precision$	$fBata$
$Frequency = 1$, $Anomaly\_ratio = 10$					
5	0.87	0.32	090	0.15	0.59
10	0.90	0.58	0.92	0.29	0.71
15	**0.91**	**0.61**	**0.92**	**0.31**	0.72
$Frequency = 10$, $Anomaly\_ratio = 7$					
30	0.93	0.82	0.93	0.41	0.82
50	0.94	**0.93**	0.94	0.46	0.86
80	0.94	0.85	0.94	**0.47**	0.86
$Frequency = 100$, $Anomaly\_ratio = 6$					
100	0.94	0.78	0.94	0.44	0.80
130	0.94	**0.91**	0.94	0.47	**0.86**
150	0.93	0.80	0.94	0.44	0.82

TABLE I: Anomaly transformer results with respect to window size and frequency

B. COPOD

For testing the algorithm we used the [7] library. we apply this algorithm with outlier probabilities of 0.5, 0.1, 0.05, 0.01, 0.005, 0.001.

1) frequency 1 Hz:

- Looking at the Outlier Frequency column, it shows the proportion of outliers in the dataset. As the frequency decreases, we observe an improvement in the model's accuracy. For instance, with an Outlier Frequency of 0.5000, the Accuracy is only 0.1923. However, as the Outlier Frequency decreases to 0.0010, the Accuracy increases significantly to 0.9609.
- Furthermore, the Recall metric measures the model's ability to identify true positives correctly, while Specificity indicates its proficiency in correctly identifying true negatives. In this dataset, with decreasing Outlier Frequency, the Recall score decreases, suggesting that the model becomes less effective in detecting all actual outliers. Conversely, the Specificity score increases, indicating a higher capability of correctly identifying non-outlier data points.
- The Precision metric signifies the model's accuracy in correctly identifying positives, while the fBeta-score combines Precision and Recall to assess the overall model performance. In this dataset, Precision shows a slight variation, but not significant improvements are observed. However, the fBeta-score increases from 0.1846 to 0.5048 as the Outlier Frequency decreases from 0.5000 to 0.0010. This indicates that the model achieves better balance between Precision and Recall at the lower Outlier Frequencies.
- In summary, the data highlights the influence of Outlier Frequency on the performance of the COPOD model. As the dataset contains fewer outliers, the model exhibits higher Accuracy and Specificity, while the Recall and fBeta-score decrease. This information helps us understand the behavior and effectiveness of the COPOD model under different settings and assists in making informed decisions on outlier detection tasks.

according to table IIAs the outlier frequency decreases, we see improvements in accuracy, precision, and recall. we obtain the

best beta score in 0.05 .

Outlier Freq	Accuracy	Recall	Specificity	Precision	fBeta-score
0.5000	0.1923	0.9500	0.1644	0.0402	0.1846
0.1000	0.7550	0.4667	0.7656	0.0683	0.5077
0.0500	0.8467	0.3333	0.8656	0.0837	0.5469
0.0100	0.9367	0.0333	0.9699	0.0392	0.5016
0.0050	0.9497	0.0167	0.9840	0.0370	0.4994
0.0010	0.9609	0.0167	0.9957	0.1250	0.5048

TABLE II: COPOD with frequency 1 Hz

Outlier Freq	Accuracy	Recall	Specificity	Precision	fBeta-Score
0.5000	0.2868	0.9190	0.2633	0.0442	0.2470
0.1000	0.7936	0.3306	0.8108	0.0609	0.5068
0.0500	0.8710	0.1884	0.8963	0.0632	0.5223
0.0100	0.9415	0.0661	0.9740	0.0862	0.5208
0.0050	0.9513	0.0446	0.9850	0.0993	0.5156
0.0010	0.9616	0.0116	0.9968	0.1186	0.5021

TABLE III: COPOD with frequency 10 Hz

2) frequency 10 Hz: according to table III As the outlier frequency decreases, we see improvements in accuracy, precision, and recall. we obtain the best beta score in 0.01 .

Outlier Freq	Accuracy	Recall	Specificity	Precision	fBeta-Score
0.5000	0.2875	0.9076	0.2646	0.0435	0.2463
0.1000	0.8184	0.2717	0.8386	0.0584	0.5094
0.0500	0.8917	0.1593	0.9187	0.0674	0.5266
0.0100	0.9501	0.0489	0.9833	0.0975	0.5170
0.0050	0.9569	0.0298	0.9911	0.1103	0.5103
0.0010	0.9633	0.0065	0.9986	0.1429	0.4998

TABLE IV: COPOD with frequency 100 Hz

3) frequency 100 Hz: according to table IV we can see the best beta- score is achived by 0.05 density of outliers.

4) frequency 200 Hz: according to table V we can see the best beta- score is achived by 0.05 density of outliers.

Outlier Freq	Accuracy	Recall	Specificity	Precision	fBeta-Score
0.5000	0.2917	0.9076	0.2690	0.0438	0.2492
0.1000	0.8204	0.2717	0.8406	0.0592	0.5108
0.0500	0.8922	0.1589	0.9193	0.0677	0.5268
0.0100	0.9502	0.0471	0.9835	0.0955	0.5162
0.0050	0.9571	0.0283	0.9914	0.1076	0.5096
0.0010	0.9633	0.0076	0.9985	0.1585	0.5004

TABLE V: COPOD with frequency 200 Hz

Window Size	Accuracy	Recall	Specificity	Precision	fBeta-score
100	0.9337	0.4884	0.9501	0.2654	0.6871
200	0.9406	0.5217	0.9561	0.3049	0.7089
400	0.9533	0.3563	0.9754	0.3481	0.6651
600	0.9607	0.0705	0.9935	0.2869	0.5355
800	0.9674	0.1383	0.9980	0.7176	0.5785
1000	0.9659	0.0705	0.9989	0.7049	0.5391

TABLE VI: COPOD with frequency 200 Hz and probability of 0.05 in different window size

Based on the data in table VI, we can observe the following trends: The window size of 200 has the highest fBeta-score of 0.7089, followed closely by the window size of 100 with a score of 0.6871. The fBeta-scores for window sizes 400, 600, and 800 are 0.6651, 0.5355, and 0.5785, respectively. The window size of 1000 has the lowest fBeta-score of 0.5391. it can concluded that the window size of 200 performs the best in terms of fBeta-score among the provided window sizes.

C. ECOD

for testing the algorithem we used the [7]libary. we apply this algorithm with outlier probabilities of 0.5, 0.1, 0.05, 0.01, 0.005, 0.001.

Outlier Prob	Accuracy	Recall	Specificity	Precision	fBeta-score
0.5000	0.2373	0.9500	0.2110	0.0424	0.2153
0.1000	0.6769	0.6500	0.6779	0.0691	0.4826
0.0500	0.7604	0.4667	0.7712	0.0698	0.5116
0.0100	0.8698	0.1167	0.8975	0.0402	0.4974
0.0050	0.8988	0.1000	0.9282	0.0488	0.5090
0.0010	0.9515	0.0667	0.9840	0.1333	0.5273

TABLE VII: ECOD with frequency 1 Hz

1) frequency 1 Hz: according to TableVII the fallowing information can be observed:

- Accuracy: The accuracy values range from 0.2373 to 0.9515. The highest accuracy is achieved for an outlier probability of 0.0010, indicating a good overall performance in predicting outliers.
- Recall: The recall values vary from 0.0667 to 0.9500. Higher recall values indicate a better ability to correctly identify outliers, with the highest recall achieved at an outlier probability of 0.5000.
- Specificity: The specificity values range from 0.2110 to 0.9840. Higher specificity values imply a better ability to accurately predict non-outliers, with the highest specificity achieved for an outlier probability of 0.0010.
- Precision: Precision values range from 0.0402 to 0.1333. Higher precision indicates a better ability to accurately classify predicted outliers, with the highest precision observed at an outlier probability of 0.0010.
- fBeta-score: The fBeta-scores range from 0.2153 to 0.5273. The fBeta-score considers both precision and recall, allowing for a balanced evaluation of outlier detection performance. The highest fBeta-score is achieved at an outlier probability of 0.0010.

for this frequency first we can see that the best probability of outliers is 0.001 . in the next step we saw the effect of window size with the use of evaluation in the last section.

Upon comparing the data using only the window size of the ECOD model, we can observe a trend in your provided data.

we conduct the test on set 5, 15, 20, 25 window size as it can be seen in TableVII As window size increases, the model's Accuracy remains relatively constant, fluctuating very slightly around a value of 0.96. The Precision, Recall, and fBeta-score, however, show considerable differences at a window size of 5, but then becomes 0.0000 from window sizes of 15 and onwards.

It's important to note the Precision, Recall, and fBeta-score were particularly higher at window size 5 compared to other window sizes, which suggests that smaller window sizes might be more effective with the ECOD model for precision, recall and fBeta-score.

2) frequency 10 Hz: The table VIII provides a summary of the evaluation metrics for the 'ECOD' model under different data frequencies, outlier frequencies,. The metrics include Accuracy, Recall, Specificity, Precision, and fBeta-score.

- At an outlier frequency of 0.5000, the model achieves an Accuracy of 0.3173, indicating that it correctly predicts

Outlier Freq	Accuracy	Recall	Specificity	Precision	fBeta-score
0.5000	0.3173	0.9190	0.2950	0.0461	0.2676
0.1000	0.7531	0.4347	0.7649	0.0642	0.5005
0.0500	0.8345	0.2876	0.8548	0.0685	0.5255
0.0100	0.9350	0.1207	0.9652	0.1141	0.5425
0.0050	0.9489	0.0959	0.9806	0.1547	0.5408
0.0010	0.9627	0.0612	0.9961	0.3700	0.5317

TABLE VIII: ECOD with frequency 10 Hz

the majority class about 31.73 of the time. The Recall is high (0.9190), suggesting a good ability to correctly identify the positive class. However, the Precision is low (0.0461), indicating a high number of false positives. The fBeta-score is 0.2676, which takes into account both precision and recall, with a higher value indicating a better balance between the two.

- As the outlier frequency decreases to 0.0010, the model's performance improves. The Accuracy increases to 0.9627, indicating better overall prediction accuracy. However, the Recall and Precision decrease, reflecting a trade-off between correctly identifying the positive class and controlling false positives. The fBeta-score also decreases slightly to 0.5317.

as are priority in this application is Beta-score there for we can sugest the outlier density is 0.01 .

3) frequency 100 Hz: The table IX show the fallowing information

- Accuracy: Accuracy measures the overall correctness of the model's predictions. As the outlier frequency decreases from 0.5000 to 0.0010, the accuracy generally improves. This suggests that the model becomes more accurate in classifying the data correctly as the outlier frequency decreases.
- Recall: Recall (also known as sensitivity) measures the ability of the model to correctly identify positive cases. In this case, it represents the model's ability to detect outliers. The recall shows a decreasing trend as the outlier frequency decreases. This means that at lower outlier frequencies, the model becomes less effective in correctly identifying true positives from the pool of positive cases.
- Specificity: Specificity measures the ability of the model to correctly identify negative cases. It shows an increasing trend as the outlier frequency decreases. This indicates that at lower outlier frequencies, the model becomes better at correctly classifying negative cases.
- Precision: Precision measures the proportion of correctly identified positive cases out of the total predicted positive cases. It shows varying behavior as the outlier frequency changes. The precision increases from 0.0461 to 0.3933 when the outlier frequency decreases from 0.5000 to 0.0010. This suggests that at lower outlier frequencies, the model becomes better at identifying true positive cases.
- fBeta-score: The fBeta-score combines precision and recall into a single metric, giving more importance to either precision or recall based on the selected beta value. As the outlier frequency decreases, the fBeta-score shows slightly varying behavior but compared to the other metrics, it remains relatively stable.

Overall, the data shows that reducing the outlier frequency tends to improve accuracy, specificity, and precision but may negatively impact recall. The fBeta-score provides a balanced measure of the model's performance by combining precision and recall.

Outlier Freq	Accuracy	Recall	Specificity	Precision	fBeta-score
0.5000	0.3388	0.8938	0.3183	0.0461	0.2797
0.1000	0.7681	0.3948	0.7818	0.0626	0.5026
0.0500	0.8486	0.2444	0.8709	0.0653	0.5233
0.0100	0.9381	0.1137	0.9685	0.1175	0.5414
0.0050	0.9520	0.0988	0.9834	0.1802	0.5444
0.0010	0.9633	0.0589	0.9967	0.3933	0.5307

TABLE IX: ECOD with frequency 100 Hz

Outlier Freq	Accuracy	Precision	Recall	fBeta-Score	Specificity
0.5000	0.3419	0.0464	0.8947	0.2819	0.3215
0.1000	0.7711	0.0634	0.3946	0.5047	0.7850
0.0500	0.8529	0.0671	0.2431	0.5261	0.8754
0.0100	0.9393	0.1225	0.1144	0.5426	0.9698
0.0050	0.9525	0.1841	0.0974	0.5441	0.9841
0.0010	0.9634	0.4008	0.0574	0.5299	0.9968

TABLE X: ECOD with frequency 200 Hz

4) frequency 200 Hz: the table X The accuracy of the models ranges from as low as 0.2700 to as high as 0.9687, depending on the chosen parameters. Precision varies from 0.0465 to a maximum of 1.0000, indicating differences in the model's ability to correctly identify positive cases. Recall rate generally remains at 1.0000, implying that the models have high sensitivity in detecting positive cases. F-score, a measure that balances precision and recall, ranges from 0.0882 to 0.1093, indicating varying degrees of overall model performance. performance of the ECOD models given the specific parameter values provided.

Window Size	Accuracy	Recall	Specificity	Precision	fBeta-score
100	0.9637	0.4224	0.9837	0.4884	0.7084
200	0.9634	0.2783	0.9886	0.4747	0.6445
400	0.9687	0.2121	0.9967	0.7005	0.6190
600	0.9667	0.0705	0.9998	0.9158	0.5397
800	0.9669	0.0705	1.0000	1.0000	0.5399

TABLE XI: ECOD with frequency 200 and probability of 0.01 in different window size

Based on the data in table XI, we can observe the following trends: As the window size increases from 100 to 200, the fBeta-score decreases slightly from 0.7084 to 0.6445. Further increasing the window size from 200 to 400, the fBeta-score continues to decrease to 0.6190. However, there is a slight increase in the fBeta-score from 400 to 600, going from 0.6190 to 0.5397. Finally, keeping the window size at 800 results in a similar fBeta-score of 0.5399 compared to the previous window size. From this analysis, we can conclude that increasing the window size does not consistently improve the fBeta-score. The best fBeta-score is observed at the window size of 100 with a value of 0.7084. Subsequent increases in the window size show either a decrease or minimal change in the fBeta-score.

D. MTAD-GAT

The tests were done for the recordings with frequency 1 Hz and 10 Hz separately. For the 1 Hz case, the model was trained for 10 epochs with the following window sizes: $[2, 4, 6, 8, 10]$. The results are reported in Table. XIII. It can be noted that the achieved results are far from optimal, the numbers suggest that not enough data has been provided to properly fit the model. It is coherent with the small dataset size for this frequency. In particular, for window size 2 and 8 the evaluation metrics are unexpectedly different than for the other window sizes. For the

TABLE XII: MTAD-GAT model, Frequency = 10 Hz

Window size	Accuracy	Precision	Recall	fBeta-score	Specificity
20	0.9440	0.4728	0.9050	0.8603	0.9461
40	0.9267	0.3984	0.8738	0.8234	0.9295
60	0.9377	0.4412	0.8514	0.8348	0.9424
80	0.9326	0.4219	0.8854	0.8361	0.9351
100	0.9296	0.4094	0.8733	0.8280	0.9326

10 Hz frequency case, the training was done over 5 epochs over the following window sizes: $[20, 40, 60, 80, 100]$, in order to maintain the proportion according to the previous frequency case. See Table. XII. This architecture provides more robust metrics throughout all the window sizes. This may suggest that the amount of training data is sufficient for this task. A smaller window size of 20 provided the best results overall. This might come from the fact that the model works well when focusing on short-term patterns or local behaviours within the time series data.

TABLE XIII: MTAD-GAT model, Frequency = 1 Hz

Window size	Accuracy	Precision	Recall	fBeta-score	Specificity
2	0.7494	0.1561	0.8857	0.6196	0.7421
4	0.9390	0.5106	0.4114	0.7024	0.9787
6	0.9433	0.4513	0.5029	0.7299	0.9670
8	0.6989	0.1393	0.9429	0.5846	0.6858
10	0.9400	0.4370	0.5943	0.7584	0.9586

V. Conclusions and future works

This report aimed to examine and evaluate state-of-the-art methods for time-series anomaly detection, namely Anomaly Transformer, MTAD-GAT, ECOD, and COPOD. The primary objective was to assess the effectiveness and applicability of these techniques on our dataset (sensor readings of a Kuka robot). The models selected belong to the unsupervised learning class, this choice being due to the absence of labels on most of our training data. This is the most frequent scenario when dealing with anomaly detection, as anomalies are expensive to label and often they consist of a very small subset of points inside a time-series. No single method emerged as universally superior in all scenarios. The choice of the most suitable method depends on the specific characteristics of the time-series data and the requirements of the application. As predicted the namely Anomaly Transformer, MTAD-GAT were not feasible for higher quantity data such as frequency 100 HZ and 200 HZ.

References

[1] J. Xu, H. Wu, J. Wang, and M. Long, "Anomaly transformer: Time series anomaly detection with association discrepancy," 2022. [Online]. Available: https://openreview.net/forum?id=LzQQ89U1qm_

[2] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," 2018.

[3] G. Rosso, "Extreme value theory for time series using peak-over-threshold method," 2015.

[4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017.

[5] H. Zhao, Y. Wang, J. Duan, C. Huang, D. Cao, Y. Tong, B. Xu, J. Bai, J. Tong, and Q. Zhang, "Multivariate time-series anomaly detection via graph attention network," 2020.

[6] H. Ren, B. Xu, Y. Wang, C. Yi, C. Huang, X. Kou, T. Xing, M. Yang, J. Tong, and Q. Zhang, "Time-series anomaly detection service at microsoft," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery &amp Data Mining*. ACM, jul 2019. [Online]. Available: https://doi.org/10.1145%2F3292500.3330680

[7] Y. Zhao, Z. Nasrullah, and Z. Li, "Pyod: A python toolbox for scalable outlier detection," *Journal of Machine Learning Research*, vol. 20, no. 96, pp. 1–7, 2019. [Online]. Available: http://jmlr.org/papers/v20/19-011.html