Machine Learning in Applications
FP01 - 2024/14 Project Report

Mustich Federico, Tcaciuc Claudiu Constantin, Tomatis Gabriele
Politecnico di Torino

CONTENTS

I Introduction 2

II Background 2
- II-A Univariate vs Multivariate context . . . 2
- II-B Type of the anomaly . . . 2
- II-C Unsupervised TAD . . . 3

III Materials and methods 3

IV Results and discussion 6
- IV-A Action Based Models . . . 6
  - IV-A1 Bayesian . . . 6
  - IV-A2 Random Forest . . . 7
- IV-B Anomaly Score Based Models . . . 7
- IV-C The importance of Thresholds . . . 9
- IV-D Results when splitting collision data . . . 10

V Conclusions and future works 10

References 11

List of Figures

1 Autoencoder architecture . . . 4
2 LSTM-AD model architecture . . . 4
3 LSTM-ED model architecture . . . 4
4 Bayesian ROC curve at 1Hz on the whole dataset . . . 6
5 Bayesian ROC curve at 10Hz on the whole dataset . . . 6
6 Bayesian ROC curve at 100Hz on the whole dataset . . . 6
7 Bayesian ROC curve at 200Hz on the whole dataset . . . 6
8 Bayesian uncertainty matrix at 1Hz on the conjunction dataset . . . 7
9 Bayesian uncertainty matrix at 200Hz on the conjunction dataset . . . 7
10 Random Forest ROC curve at 1Hz on the conjunction dataset . . . 7
11 Random Forest ROC curve at 10Hz on the conjunction dataset . . . 7
12 Random Forest ROC curve at 100Hz on the conjunction dataset . . . 7
13 Random Forest ROC curve at 200Hz on the conjunction dataset . . . 7
14 LSTM-ED ROC curve at 200Hz on the conjunction dataset . . . 8
15 LSTM-ED Precision-Recall curve at 200Hz on the conjunction dataset . . . 8
16 LSTM-ED anomalies over time, in red are the predicted ones, in blue the true anomalies registered in the collision file for the perior 14:00-14:30 . . . 8
17 LSTM-ED anomalies over time, in red are the predicted ones, in blue the true anomalies registered in the collision file for the perior 16:40-17:10 . . . 8
18 LSTM-AD ROC curve at 200Hz on the conjunction dataset . . . 9
19 RNN-EBM ROC curve at 200Hz on the conjunction dataset . . . 9
22 Isolation Forest ROC curve at 200Hz on the conjunction dataset . . . 9
20 Autoencoder ROC curve at 200Hz on the conjunction dataset . . . 9
21 DAGMM ROC curve at 200Hz on the conjunction dataset . . . 9
23 LSTM-ED anomalies detected with MAD threshold at 10Hz . . . 10
24 LSTM-ED anomalies detected with MAD threshold at 10Hz . . . 10
25 Detected collisions (in red) vs labeled collisions (in blue) for LSTM-ED on recording 1. . . . 10
26 Detected collisions (in red) vs labeled collisions (in blue) for LSTM-ED on recording 5. . . . 10

List of Tables

I Bayesian Mean accuracy for 1Hz and 200Hz . . . 7
II F1 score, accuracy, precision, and recall for all the models trained on the entire dataset, tested on the concatenation of recording 1 and 5 of the dataset, at 1 Hz . . . 8
III F1 score, accuracy, precision, and recall for all the model trained on the entire dataset, tested on the concatenation of recording 1 and 5 of the dataset, at 10 Hz . . . 8
IV F1 score, accuracy, precision, and recall for all the model trained on the entire dataset, tested on the concatenation of recording 1 and 5 of the dataset, at 100 Hz . . . 8
V F1 score, accuracy, precision, recall for all the model trained on the entire dataset, tested on the concatenation of recording 1 and 5 of the dataset, at 200 Hz . . . 8
VI anomalies detected at various threshold for LSTM-ED at frequency of 10Hz and 200Hz . . . 9
VII Anomalies detected at various thresholds for LSTM-ED at frequency of 10Hz . . . 9
VIII F1 score, accuracy, precision, and recall for all the model trained on the entire dataset, tested only on recording 1, at 10 Hz . . . 10
IX F1 score, accuracy, precision, and recall for all the model trained on the entire dataset, tested only on recording 5, at 10 Hz . . . 10

Evaluating State-of-the-Art Models for Unsupervised Collision Detection in Industrial Robot Time Series

***Abstract*—This study presents a comparative analysis of state-of-the-art models for unsupervised anomaly detection in multivariate time series data from industrial robots. We evaluate an ad-hoc developed Bayesian model and seven other models, including deep learning approaches like LSTM-based architectures and Autoencoders, as well as tree-based methods like Random Forest and Isolation Forest, on a dataset of sensor readings from KUKA robots. The models are assessed on their ability to detect anomalous events corresponding to robot collisions into some obstacles. We examine performance across different data sampling frequencies and explore the impact of various anomaly score thresholding techniques. Our results indicate that LSTM-based models, particularly LSTM Encoder-Decoder, achieve the highest overall performance, with F1 scores up to 0.94 at 200 Hz sampling. We also find that the choice of threshold significantly affects detection results, with Median Absolute Deviation (MAD) typically providing optimal performance. This work provides insights into the strengths and limitations of different anomaly detection approaches for industrial time series data and highlights important considerations for practical implementation.**

I. Introduction

Today, the amount of data produced during everyday life is at its historical peak. From smart cities to the financial sector, from health care, every device produces several types of data. Among these data, we have signals that can be analyzed retrieving useful information. Signal analysis has application in the most disparate fields, from industry to medicine; for example, Liu et al. [1] proposed a deep learning framework that effectively classifies wireless signals in industrial cognitive radio networks, overcomes uncertainty in feature parameters, and improves spectrum sharing. Krisgnan et al. [2], instead, developed a deep learning process in the biomedical field to better understand and analyze biomedical signals. In our research, we focus on time series anomaly detection (TAD), which aims to find anomalous events over different signals and combinations of them.
Furthermore, the use of methods based on artificial intelligence in most applications is growing rapidly and the development of new models is growing at an impressive pace. This led to the introduction and improvement of several types of model architecture such as CNNs, RNNs ([3], [4]), GANs ([5], [6]), Tranformers ([7]), Tree-based ([8], [9]), Cluster-based ([10]) etc...) and approaches (self-supervised ([11]), semi-supervised ([6], [8], [12]), unsupervised [10], [13]. Most of them could be effectively employed on TAD tasks, each with their own strength and weaknesses. So, given this large variety of models and variants, following the work of Gen Li et al. [14], we propose a comparison between seven SOTA models present in the research literature.

For our research we use a dataset consisting of a KUKA industrial robot recordings, which collects data from six sensors of different type ordered by time recorded at different frequencies; in this domain the considered anomalies were found when the robot had a collision with an obstacle. The dataset is already organized into a collection of "normal" recordings, containing no collisions, and two recordings containing anomalies. The models are taken among a set of different machine learning methodologies in order to better assess which category can be used the most effectively on this kind of task. The intended goal of this work is to present a comparison between different models.

For each model, we start by analyzing the dataset, apply some pre-processing techniques and trained our models on the normal data. We proceed then by evaluating their performance in detecting anomalies on the collision data, used as a test set. For each model we compute different metrics: precision, recall, F1-score, AUCs and ad-hoc graphs on the results. We will go deeper into these aspects in the next sections.
We find that Deep Learning-based models (Autoeconder and LSTM-ED in particular) perform decently on the considered task, whereas some classic machine learning approaches (DAGMM) show unreliable performances. We report in the following sections the obtained results.

II. Background

We briefly introduce here some background concepts, related to time series in general, not only to the anomaly detection task.

A. Univariate vs Multivariate context

A univariate time series [15] is a time series in which there is only one variable, x, varying over time (e.g. temperature in a room). On the other hand, multivariate time series [10], [16] are characterized by a set of signals, $\mathbf{x}$, oscillating over time.

B. Type of the anomaly

Anomalies can be classified into three main categories [14]:

- *Point anomalies* are characterized by a single point that violates a given threshold on a fixed instant. In the univariate case it could be an abnormal peak of the signal, while in a multivariate context, since we deal with multiple signals, we may consider a subset of them. In our work we define an anomaly score function (the output score of the prediction) and we consider as point

anomalies those timestamps for which the anomaly scores surpassed the considered threshold.
- *Anomaly intervals*, are similar to anomaly points, but they extend to multiple, contiguous instants. The majority of anomalies encountered within this project can be classified as belonging to this category. It is worth noting how different sampling frequency could result in an anomaly being detected as either a point anomaly or an anomalous interval. In our work we used ground truth anomaly intervals and compared the results against those.
- *Anomaly time series* are related to a signal that is considerably different from the data observed during training across its whole duration or most of it. In this case, the whole time series would be considered as abnormal.

C. Unsupervised TAD

Unsupervised TAD approaches [17] are methodologies in which the signals considered during the model training phase differ from those used during the test phase. More precisely, an unsupervised TAD model is trained **only** on the *normal* data and at test phase it should be able to assess whether the signal it is analyzing match with the desired behaviour it encountered during training. Anomaly detections may be triggered by an isolated signal exceeding a predefined threshold (see section III for details) or by some unexpected pattern among the signals which was not present in the training data. Models of this kind should handle edge cases such as in our considered data where noticeable time gaps can be present between one recording and another. In our data there was a significant two and a half hour gap between the two recordings used for collision detection at testing phase. To address this issue, a robust dataset should be curated, characterized by minimal impurities. This can be achieved through repetitive signal recording or the implementation of sophisticated feature engineering techniques prior to model training. The optimal solution is contingent upon the specific application domain.

III. Materials and methods

As already introduced, there exists a large variety of models and approaches. For our research purposes we compare seven SOTA models that were collected by Schimd et al. in [18] with the addition of a Bayesian model developed in class.

We start by analyzing the considered dataset. It consists of an ensemble of *.csv*, *.xlsx*, and *.metadata* files. Each of those containing different information:

- *.csv* files contain the recording for every sensor given a recording frequency;
- A single *.xlsx* file contains the start and the end of each anomaly that recorded. This document had two sheets related to different recording temporal windows. For every model, we considered the concatenation of this recordings and then each one by itself.
- *.metadata* files contain information such as name of an action, starting time, duration etc.;

Our initial efforts concentrated on the preprocessing of the raw files. Subsequently, we conducted statistical analyses utilizing Python notebooks, making use of widely adopted software libraries such as *numpy*, *pandas*, and *matplotlib*. Additional libraries were employed, all of which are documented in a file named *requirements.txt*. The data preprocessing involved importing the dataset into dataframes and normalizing the data. We then reviewed the existing literature for state-of-the-art (SOTA) TAD models, with the objective of identifying a minimum of four SOTA models for comparative analysis against the Bayesian model. Selection of SOTA models was guided by the goal of ensuring diversity among model types to preclude redundancy in results due to similar model typologies. Specifically, we compared the Bayesian model with seven SOTA models: *Autoencoder, Isolation Forest, LSTM-AD, LSTM-ED, Random Forest, RNN*, and *DAGMM*. Below, we offer a brief introduction to each of those:

Bayesian model The considered Bayesian model makes prediction on data by using an MLP model with a dropout at its core. This procedure is re-iterated for *n* times and then the results are averaged. On success, these results are used as a score to predict an action that will be evaluated through the confidence class, where some metrics such as entropy, variance and max softmax response are computed over the predictions. Results are expressed in form of a confusion matrix where each cell identify how many actions of that type are correctly classified showing a percentage ratio. An anomaly is detected based on how confident this model is on predicting a certain action, if the classifier is not confident over a certain prediction, we label that prediction as an anomaly. This model was developed during the laboratories in class.

Autoencoder[1] An autoencoder is a kind of neural network architecture that is trained to compress and reconstruct data distributions, making it well suited for unsupervised anomaly detection in multivariate time series. It consists of an encoder network $E(x)$ that maps input x to a lower-dimensional latent space z, and a decoder network $D(z)$ that attempts to reconstruct the original input from z. The model is trained to minimize the reconstruction error $L(x, x')$, typically using mean squared error. For time series anomaly detection, a typical autoencoder is trained on normal, non-anomalous data. It learns to effectively reconstruct typical patterns in the time series. During inference, we compute the reconstruction error for each time step. Anomalies are identified when the reconstruction error exceeds a threshold τ. In our multivariate setting, x_t represents the input of d-dimensional in time t.
Advantages of autoencoders include their ability to capture complex non-linear relationships in multivariate data and their unsupervised nature, which is crucial when labeled anomalies are scarce. They can adapt to various types of anomalies without explicit specification and handle high-dimensional input effectively.
However, autoencoders have limitations. They may struggle with capturing long-term temporal dependencies, potentially missing anomalies that manifest over extended periods. Setting an appropriate threshold τ can be challenging and may require domain expertise or additional validation data.

Moreover, autoencoders might be sensitive to noise in the training data, potentially learning to reconstruct anomalous patterns if present in the training set.

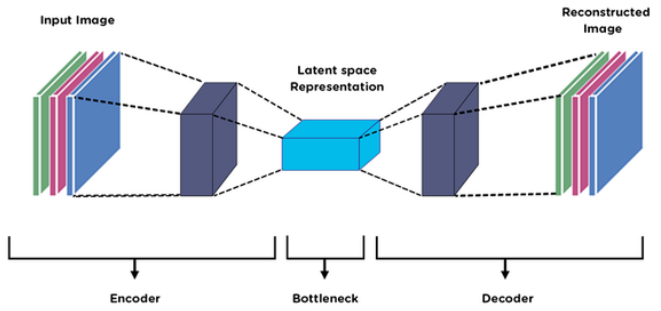

Fig. 1. Autoencoder architecture

Isolation Forest This is a multivariate tree-based model [9] suited for outlier detection. We took the implementation from *sklearn.ensemble*. This algorithm isolates observations by randomly selecting a feature and splitting choices based on a random value included in the feature domain. Several trees are built using different features and samples are classified in each of them. Then an averaged path for every prediction is computed overall. Anomalies tend to have shorter path and could be detected in this way by the model. Moreover, we made a little parameters selection via fine-tuning on *n_estimators* and on the *contamination* parameters. *n_estimators* indicates how many trees are built, while *contamination* is the proportion of anomalies with respect to normal points. The Isolation Forest algorithm demonstrates strong applicability for outlier detection tasks and represents a strong candidate for our specific problem.

LSTM-AD[1] LSTM-AD (Long Short-Term Memory for Anomaly Detection) is a specialized application of LSTM networks for multivariate time series anomaly detection, specifically engineered to identify anomalous patterns in time-dependent data unlike traditional LSTM models used for sequence prediction or classification.
During inference, anomaly detection is performed by comparing the prediction error with a threshold τ. This approach allows LSTM-AD to learn complex patterns of normal behavior and subsequently identify deviations as potential anomalies. The model's ability to capture long-term dependencies makes it particularly effective in detecting contextual and collective anomalies that may not be apparent in isolated time steps.
Although LSTM-AD offers robust performance in capturing temporal patterns, it faces challenges such as high data requirements and sensitivity to hyperparameters. In addition, the selection of an appropriate anomaly threshold τ remains a critical consideration.

[1]All of the implementation of these models was taken from https://github.com/KDD-OpenSource/DeepADoTS

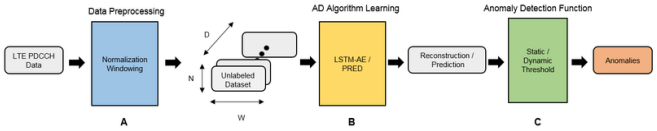

Fig. 2. LSTM-AD model architecture

LSTM-ED[1] LSTM-ED (Long Short-Term Memory Encoder-Decoder) represents an architecture for multivariate time series anomaly detection, combining the sequential modeling capabilities of LSTMs with the reconstruction principle of autoencoders. LSTM-ED is trained on normal data to minimize the reconstruction error.
Anomaly detection is performed once again using a threshold-based approach to the reconstruction error.
The key aspect of LSTM-ED lies in its ability to capture both the encoding of the entire input sequence into a fixed-length vector and the decoding of this representation back into a sequence. This allows the model to learn complex, nonlinear relationships in the temporal data while maintaining the ability to reconstruct the entire sequence.
LSTM-ED excels at detecting anomalies that manifest as unusual patterns over extended periods, which makes it particularly suitable for identifying collective anomalies. The encoder-decoder structure enables the model to consider the full context of a sequence when determining anomalies, potentially leading to more nuanced detection compared to single-directional LSTM models.
However, LSTM-ED shares some limitations with other deep learning approaches, including high computational requirements, potential overfitting on small datasets, and the selection of an appropriate anomaly threshold.

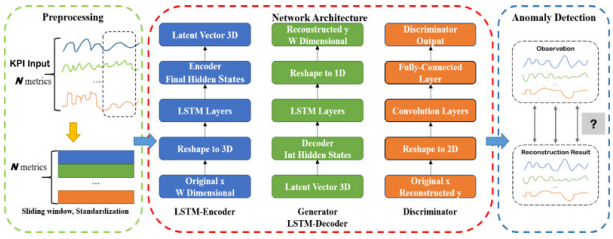

Fig. 3. LSTM-ED model architecture

Random Forest Random forest is a semi-supervised, tree-based algorithm that constructs multiple decision trees over distinct sub-samples of the dataset, subsequently averaging the results to enhance accuracy and mitigate overfitting. This methodology is similar to that of the Bayesian model, wherein the predictions generated represent action scores rather than anomaly scores. While efforts were made to fine-tune the parameters, the process was terminated due to the prohibitive number of parameter combinations (exceeding 200), which resulted in an extraordinarily slow training

phase as it necessitated the training of over 200 classifiers. Consequently, parameters closely aligned with the default settings of the *sklearn* model were employed. Although random forest constitutes a straightforward classifier with an easily interpretable structure, its efficacy may be significantly contingent upon the specific application.

RNN-EBM[1] The Recurrent Neural Network Energy-Based Model (RNN-EBM) represents an attempt to combine the sequential modeling capabilities of RNNs with the probabilistic framework of energy-based models. The RNN-EBM's distinctive feature is its ability to learn a probability distribution over sequences implicitly through an energy function. This allows the model to capture complex temporal dependencies and multimodal patterns in the data, potentially leading to more robust anomaly detection compared to traditional reconstruction-based approaches. The energy-based formulation provides a principled way of scoring the likelihood of sequences, offering a suitable architectural option for anomaly detection. Moreover, the RNN component enables the model to capture long-range dependencies in the time series data. Anomaly detection is performed again by comparing the energy of a sequence to a threshold.

DAGMM[1] Deep Autoencoding Gaussian Mixture Model (DAGMM) in the ensemble of an autoencoder and a GMM model. The autoencoder is used to obtain a lower dimensional input space that can be well approximated by the GMM model. During the reduction of the feature space a reconstruction error is computed; this will be used by the GMM to parameters estimations and to predict the belonging or not to a certain GMM. DAGMM showed high performances on public benchmarks, but, despite that, showed worst results in our case if compared with other models.

XGBoost In our implementation, we adapted XGBoost (Extreme Gradient Boosting) [19] for multivariate time series anomaly detection using an autoencoder-like approach. XGBoost is an ensemble of decision trees. The model takes a multivariate time series input and attempts to output the same values. This process is the same of the encoding and decoding phases of an autoencoder. The anomaly detection mechanism relies on the reconstruction error. After training the model on normal data, we use it to reconstruct inputs during the inference phase. The reconstruction error is calculated as the difference between the original input and the reconstructed output. Anomalies are identified when this reconstruction error exceeds a predetermined threshold. This approach leverages XGBoost's ability to capture complex, non-linear relationships within the data while adopting the intuition behind autoencoder-based anomaly detection. When compared with other considered detection models, we find that XGBoost requires surprisingly more training time and computational resources (around 3 hours for training). For this reason, we decided to discard this model.

With each trained model, we compute some statistics and graphs that will be explained more specifically in the next section. More in general outputs are different for two major class of models, depending on what the prediction is.

Bayesian model and the random forest output a number between [0; 31] that is referring to an **action prediction** (e.g. *picking object from pallet x*, *moving from pallet n to m*, etc...) that has to be evaluated through a measure of a **confidence**. This is a Python class developed during our laboratories and aims to find those actions on which the model is not so sure about. The thought here was that if the model is not so confident about a predicted action, maybe an anomaly occurred when that action happened. So in this case, the accuracy has been done comparing actions. Here we plotted confusion matrices showing the relation between the predicted action and the actual one; histograms showing entropy, variance and max softmax response for correct, wrong and all action predictions; ROC curves for the cited metrics. This choice was made in order to keep the model as similar as the bayesian one, in order to do better comparison between these similar algorithms.

On the other hand, other models, instead, output directly the anomaly scores from which we can derive the *anomaly prediction* (targeted as 0 and 1 or -1 and 1 depending on the model). So, basically, we had to find when the movement of the robot (stored for example in *rec1_collision_20220811_rbtc_0.01s.csv*) hit an object, leading to a collision (stored in *20220811_collisions_timestamp.xlsx*). So, we built a dataframe with the various collision zones extracted from the excel file and we set as *is_collision* all the timestamps related to the signals that were inside that temporal window having a sort of ground truth labeling. Finally, we compared that ground truth with the prediction and we computed analysis metrics. The predictions were made by comparing the anomaly scores with respect to different thresholds. We used five major thresholds during our research:

- *STD*: this threshold is set to two standard variance over the mean.
$$std\_thresh = \mu + 2\sigma$$
- *MAD*: this threshold is set to 2 MAD over the median.
$$mad\_thresh = median + 2MAD$$
$$MAD = median\left(|X - median|\right)$$
- *Percentile*: this threshold is set to the 95th data percentile. This means that 5% of data will be targeted as potential anomaly.
$$thresh\_p_{95} = P_{95}$$
- *IQR*: this threshold is set to 1.5 times the IQR over the third quartile.
$$Q_1 = P_{25}$$
$$Q_3 = P_{75}$$
$$IQR = Q_3 - Q_1$$
$$thresh_{IQR} = Q_3 + 1.5 * IQR$$
- *Zero*: anomaly scores were compared with 0 as threshold to obtain the related anomaly prediction; this was used only in the isolation forest as this model output anomaly

scores that are more negative where it is most sure that a certain instant is an anomaly.

As said before, for all the models except isolation forest, a classification based on different threshold was made and the results were interesting. With this type of approaches we computed metrics such as F1-score, accuracy, precision and recall; we plotted an AUC-ROC and an AUC-PR curve for each configuration of each model; a scatter plot showing the chosen threshold, the corresponding detected anomalies and the related anomaly score; finally, we plot a graph with true collisions zones vs the predicted ones.

For each model, after computing the analysis, we also checked what would have been the best threshold that maximizes the F1-score value, to see how close was our choice was with respect to the optimal threshold for the test dataset. Related to this we have to notice that as demonstrated by Kim et al. [17] F1-score could be overestimated with by implementing the point adjustment method for treating anomalies. We want to clarify that we do not use this methodology in our project, maintaining the collisions as they are in the original dataset, without extending any instant into a window.

IV. Results and Discussion

In this section we will go deeply into the obtained results from the different models. Starting from the dataset, we already know how the data are splitted among the different files. We now focus on some important aspects we noticed. First of all we have to say that for a time discrepancy the time interval of all the collisions had to be anticipated by two hours. Then we have some collisions only during the recording 1 and 5 as the name of the two sheet in *20220811_collisions_timestamp.xlsx* may suggest. Moreover, during our work, we found a huge gap in this file in the "rec1" sheet; in fact, between 16:29:46 and 19:04:44 there are no anomalies recorded; anomaly recording's resumes after 19:04:44 and goes until 19:09:58. But, if we examine our collision test dataset, recordings are stopped around 14:30 (corresponding to 16:30 in the excel file), so every model will never predict anomalies in the last five minutes. This could cause some troubles if we consider only "rec1" during our analysis, but it does not seem a problem using the conjunction of the recordings, since this gap will be eliminated by incorporating the "rec5" in the considered temporal window. Finally, we have to say that we decided to split the analysis also for "rec1" and "rec5" in standalone configuration; this was made in order to assert that the conjunction of the two dataset would not lead to fake anomalies being detected in that area. Now, let us consider the obtained results.

A. Action Based Models

This models, rather than classifying anomalies directly, output labels that are related to the action performed by the various robots and the anomaly detection is performed indirectly by using a confidence class which job is to assert how confident the classifier is on a prediction.

1) Bayesian: We first asses the performance of the Bayesian model; to do so we used the ROC curve for the models that predicts the anomaly through a confidence metrics.

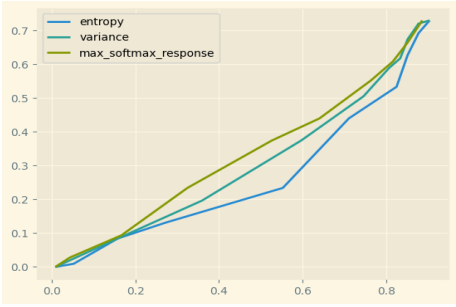

Fig. 4. Bayesian ROC curve at 1Hz on the whole dataset

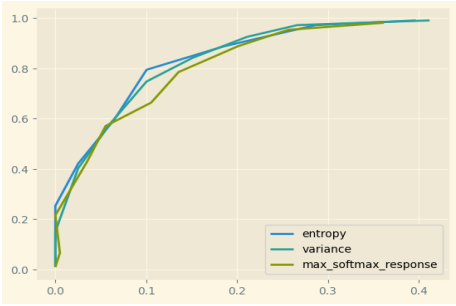

Fig. 5. Bayesian ROC curve at 10Hz on the whole dataset

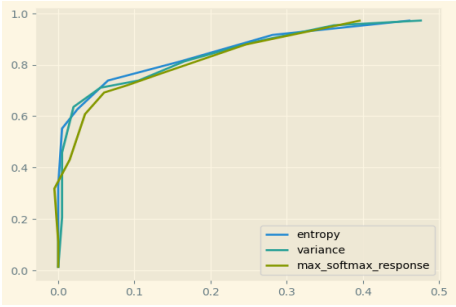

Fig. 6. Bayesian ROC curve at 100Hz on the whole dataset

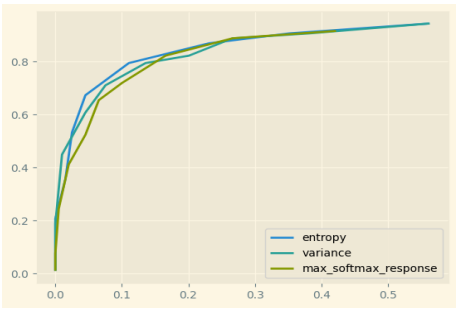

Fig. 7. Bayesian ROC curve at 200Hz on the whole dataset

As we can see from 4, 5, 6, 7 the performance at a low frequency is not optimal at all, but, as the frequency increases

and more data are available, the performance increases and it reaches its maximum at 200Hz.

We can further analyze the best scenario (200Hz) and the worst scenario (1Hz); firstly, since this models predicts action, we want to see how good this model is into making predictions.

TABLE I
BAYESIAN MEAN ACCURACY FOR 1HZ AND 200HZ

Frequency	Mean accuracy
1 Hz	30.0 ± 25.3
200 Hz	74.8 ± 16.0

What we see from I is that at low frequency the model is not even good at detecting the action performed by the robot at a certain time, while with high frequency the model has a fairly good accuracy.

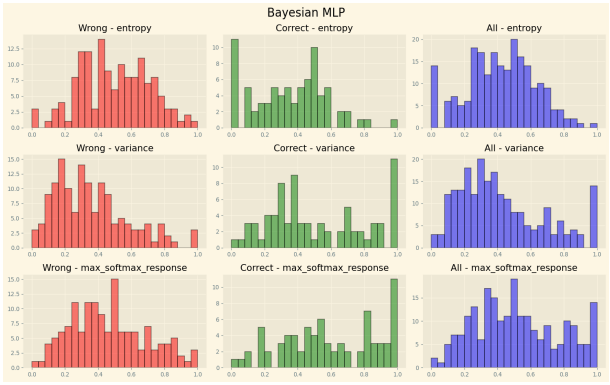

Fig. 8. Bayesian uncertainty matrix at 1Hz on the conjunction dataset

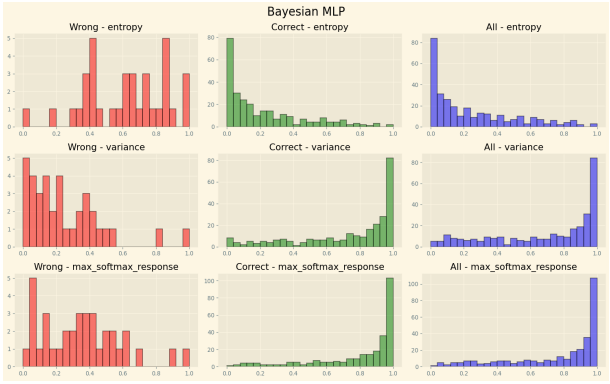

Fig. 9. Bayesian uncertainty matrix at 200Hz on the conjunction dataset

The uncertainty plot demonstrate 8 how the model is very unsure on his prediction at low frequency, but it is quite confident on the predictions made at a higher frequency 9.

2) Random Forest: A similar model that behave like the Bayesian one is the Random forest. The similarity comes from the fact the both of them predict actions and not anomaly, and the anomaly is asserted based on a confidence metric.

The performance of this model is superior to the performance of the Bayesian model; only at low frequency its behaviour is discrete, for higher frequency the performance is better. 10, 11, 12, 13

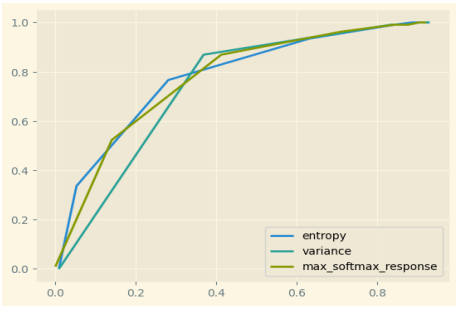

Fig. 10. Random Forest ROC curve at 1Hz on the conjunction dataset

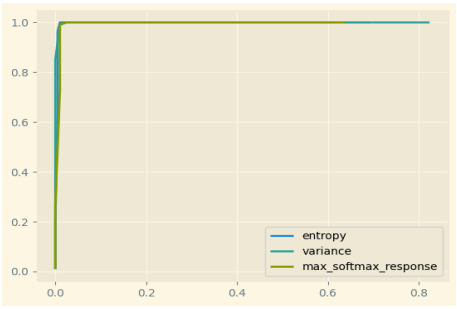

Fig. 11. Random Forest ROC curve at 10Hz on the conjunction dataset

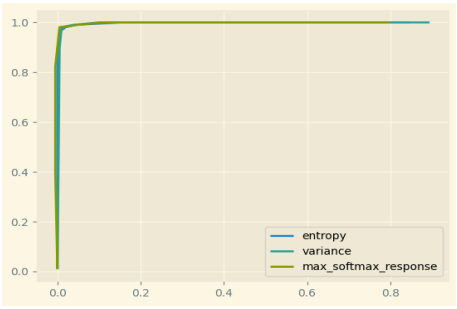

Fig. 12. Random Forest ROC curve at 100Hz on the conjunction dataset

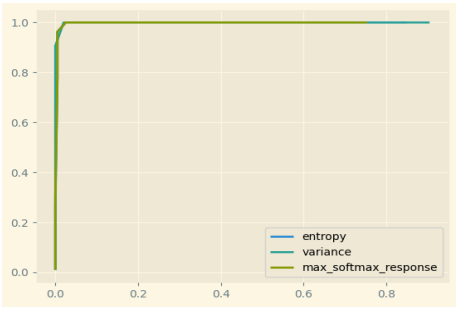

Fig. 13. Random Forest ROC curve at 200Hz on the conjunction dataset

B. Anomaly Score Based Models

This models outputs are scores that can be directly used to predict an anomaly based on a threshold. As mentioned

before, for these models we asserted their performance through different scores, such as F1-score, but we focused mainly on Precision and Recall.

For the following results, we used MAD as threshold for our measurements, because we found being the optimal one among the others and this is also addressed by Leys et al. [20] that studied it with respect to the outlier detection task.

TABLE II
F1 SCORE, ACCURACY, PRECISION, AND RECALL FOR ALL THE MODELS TRAINED ON THE ENTIRE DATASET, TESTED ON THE CONCATENATION OF RECORDING 1 AND 5 OF THE DATASET, AT 1 HZ

	F1 score	Accuracy	Precision	Recall
LSTM-ED	0.8308	**0.8507**	0.9000	0.7714
LSTM-AD	0.8168	0.8416	**0.9070**	0.7429
RNN-EBM	0.5521	0.6697	0.7759	0.4286
Autoencoder	0.7416	0.7919	0.9041	0.6286
DAGMM	0.0000	0.5249	0.0000	0.0000
Isolation-Forest	**0.8584**	0.8507	0.7812	**0.9524**

TABLE III
F1 SCORE, ACCURACY, PRECISION, AND RECALL FOR ALL THE MODEL TRAINED ON THE ENTIRE DATASET, TESTED ON THE CONCATENATION OF RECORDING 1 AND 5 OF THE DATASET, AT 10 HZ

	F1 score	Accuracy	Precision	Recall
LSTM-ED	**0.9333**	**0.9510**	0.8570	**1.0000**
LSTM-AD	0.8252	0.8824	0.8416	0.8095
RNN-EBM	0.8531	0.8987	0.8491	0.8571
Autoencoder	0.8571	0.9020	0.8571	0.8571
DAGMM	0.2484	0.6046	0.3571	0.1905
Isolation-Forest	0.9140	0.9379	**0.8707**	0.9616

TABLE IV
F1 SCORE, ACCURACY, PRECISION, AND RECALL FOR ALL THE MODEL TRAINED ON THE ENTIRE DATASET, TESTED ON THE CONCATENATION OF RECORDING 1 AND 5 OF THE DATASET, AT 100 HZ

	F1 score	Accuracy	Precision	Recall
LSTM-ED	**0.9375**	**0.9542**	**0.8824**	**1.0000**
LSTM-AD	0.8431	0.8954	0.8687	0.8190
RNN-EBM	0.8744	0.9118	0.8545	0.8952
Autoencoder	0.9327	0.9510	0.8814	0.9905
DAGMM	0.0000	0.6569	0.0000	0.0000
Isolation-Forest	0.9083	0.9346	0.8761	0.9429

TABLE V
F1 SCORE, ACCURACY, PRECISION, RECALL FOR ALL THE MODEL TRAINED ON THE ENTIRE DATASET, TESTED ON THE CONCATENATION OF RECORDING 1 AND 5 OF THE DATASET, AT 200 HZ

	F1 Score	Accuracy	Precision	Recall
LSTM-ED	**0.9417**	**0.9575**	**0.8898**	**1.0000**
LSTM-AD	0.8021	0.8758	0.8851	0.7333
RNN-EBM	0.8704	0.9085	0.8468	0.8952
Autoencoder	0.8899	0.9183	0.8279	0.9619
DAGMM	0.0177	0.6373	0.1250	0.0095
Isolation-Forest	0.9123	0.9346	0.8455	0.9905

From the tables II, III, IV-B, V, We can see that the DAGMM model is the one performing worse among this models, obtaining some noticeable results only at a frequency of 10Hz. The model that achieves the best score overall is LSTM-ED, performing the top score in every category at a frequency of 200Hz, achieving good results also for different frequencies. To further asses the performance of this model we also plot the AUC-ROC 14 and AUC-PR 15 curves.

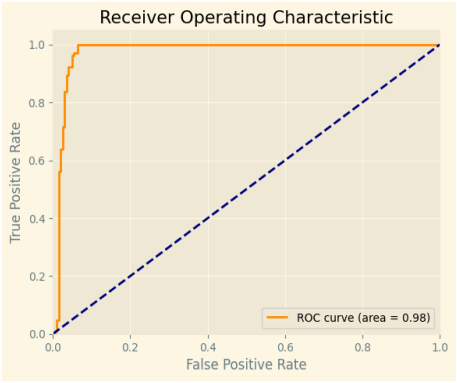

Fig. 14. LSTM-ED ROC curve at 200Hz on the conjunction dataset

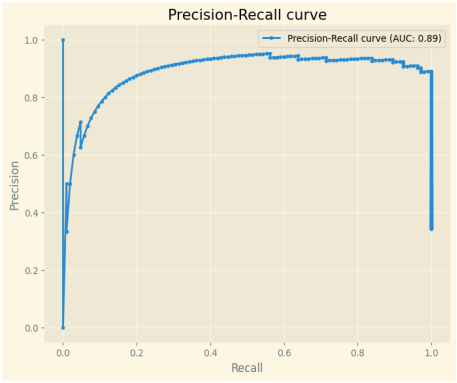

Fig. 15. LSTM-ED Precision-Recall curve at 200Hz on the conjunction dataset

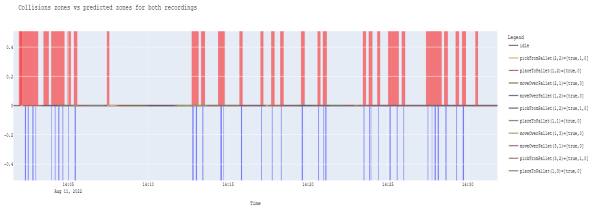

Fig. 16. LSTM-ED anomalies over time, in red are the predicted ones, in blue the true anomalies registered in the collision file for the perior 14:00-14:30

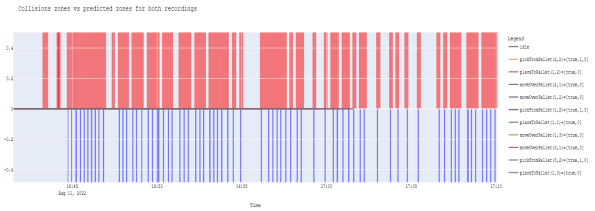

Fig. 17. LSTM-ED anomalies over time, in red are the predicted ones, in blue the true anomalies registered in the collision file for the perior 16:40-17:10

As we can see from figure 16 and 17 the predicted anomalies match the true collisions.

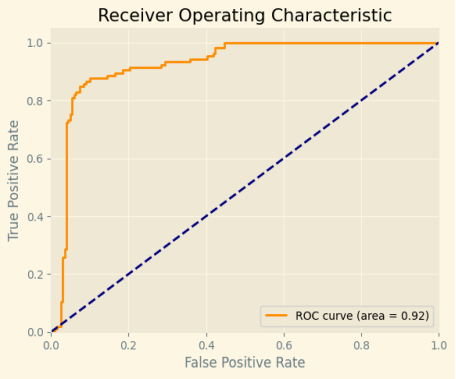

Fig. 18. LSTM-AD ROC curve at 200Hz on the conjunction dataset

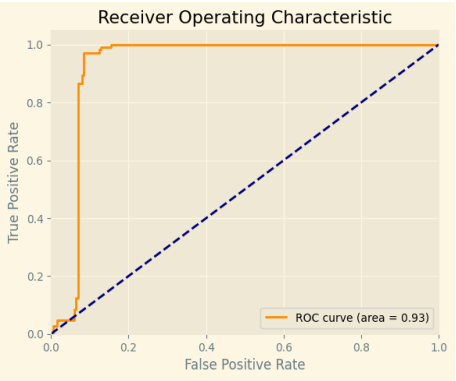

Fig. 19. RNN-EBM ROC curve at 200Hz on the conjunction dataset

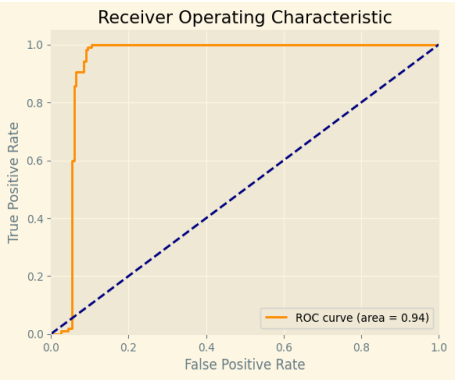

Fig. 22. Isolation Forest ROC curve at 200Hz on the conjunction dataset

For a more comprehensive read of the results we also report ROC for the remaining models at a frequency of 200Hz 18, 19, 20, 21 and 22.

C. The importance of Thresholds

Now we dive into the importance of choosing the best threshold for the anomaly score based models. To do so we will focus on data with a frequency of 10Hz and 200Hz and we will use LSTM-ED model.

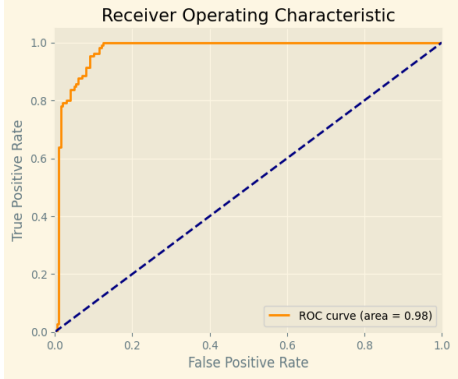

Fig. 20. Autoencoder ROC curve at 200Hz on the conjunction dataset

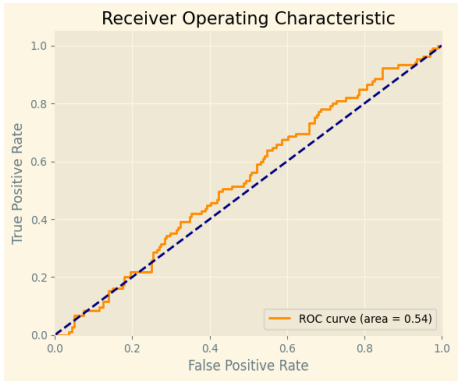

Fig. 21. DAGMM ROC curve at 200Hz on the conjunction dataset

TABLE VI
ANOMALIES DETECTED AT VARIOUS THRESHOLD FOR LSTM-ED AT FREQUENCY OF 10HZ AND 200HZ

Frequecy	STD	MAD	Percentile	IQR
10 Hz	12	120	16	7
200 Hz	1	118	16	6

As we can see from VI and 23, 24 the selection of the threshold deeply impacts the number of anomalies our model detects; this is more evident at higher frequency for the presence of very high score outputted by the model (LSTM-ED in our case, but this trends can also be seen in the other models).

To further asses the correctness of our threshold VII, we also computed the best possible threshold, that would produce the highest F1-score and, in all cases, MAD was the one that gave us the closest results.

TABLE VII
ANOMALIES DETECTED AT VARIOUS THRESHOLDS FOR LSTM-ED AT FREQUENCY OF 10HZ

Threshold	Perfect	STD	MAD	Percentile	IQR
LSTM-AD	47	8	52	9	18
LSTM-ED	119	10	120	16	7
RNN-EBM	128	14	106	16	14
Autoencoder	116	11	105	16	16
DAGMM	304	13	56	16	0

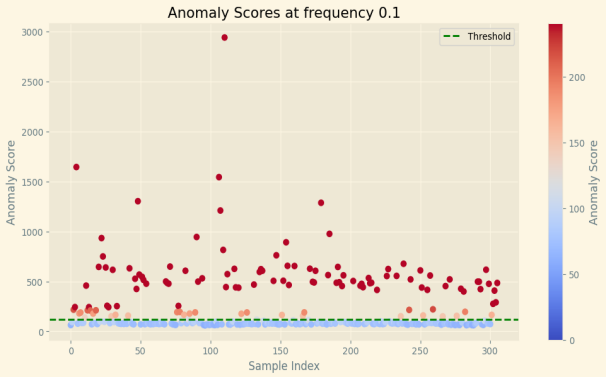

Fig. 23. LSTM-ED anomalies detected with MAD threshold at 10Hz

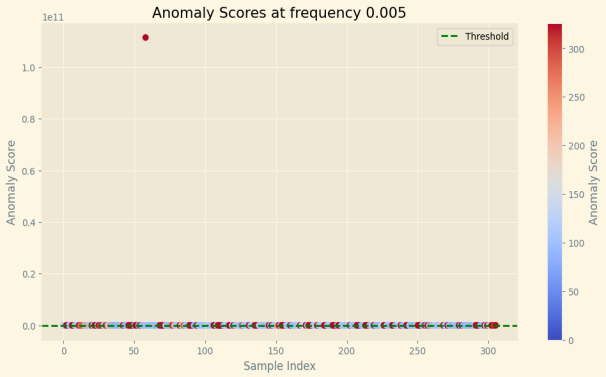

Fig. 24. LSTM-ED anomalies detected with MAD threshold at 10Hz

D. Results when splitting collision data

As previously stated, the collision dataset considered in this work consists of a collection of two different robot activity recordings. So far, we presented results on the comprehensive dataset, obtained by concatenating the two recordings. However, one might infer that concatenating two separate activity logs from different robot actions, separated in time by a considerable margin, might result in false positive anomalies detected on the region corresponding to the merged section between the twos. For this reason, we conducted our experiments also on each of these two recordings separately. The results, for sampling frequency of 10 Hz only, are reported in VIII and IX.

TABLE VIII
F1 SCORE, ACCURACY, PRECISION, AND RECALL FOR ALL THE MODEL TRAINED ON THE ENTIRE DATASET, TESTED ONLY ON RECORDING 1, AT 10 HZ

	F1 score	**Accuracy**	**Precision**	**Recall**
LSTM-ED	**0.8537**	0.9268	0.7447	**1.0000**
LSTM-AD	0.7126	0.8476	0.5962	0.8857
RNN-EBM	0.8500	0.9268	0.7556	0.9714
Autoencoder	0.7816	0.8841	0.6538	0.9714
DAGMM	0.2985	0.7134	0.3125	0.2875
Isolation-Forest	0.8732	**0.9451**	**0.8611**	0.8857

TABLE IX
F1 SCORE, ACCURACY, PRECISION, AND RECALL FOR ALL THE MODEL TRAINED ON THE ENTIRE DATASET, TESTED ONLY ON RECORDING 5, AT 10 HZ

	F1 score	**Accuracy**	**Precision**	**Recall**
LSTM-ED	0.7447	0.8298	**0.9211**	0.6250
LSTM-AD	0.2941	0.6596	0.8333	0.1786
RNN-EBM	0.0606	0.5603	0.2000	0.0357
Autoencoder	0.3889	0.6879	0.8750	0.2500
DAGMM	0.1370	0.5532	0.2941	0.0893
Isolation-Forest	**0.8296**	**0.8369**	0.7089	**1.000**

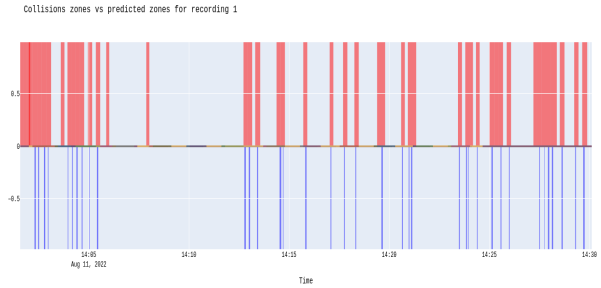

Fig. 25. Detected collisions (in red) vs labeled collisions (in blue) for LSTM-ED on recording 1.

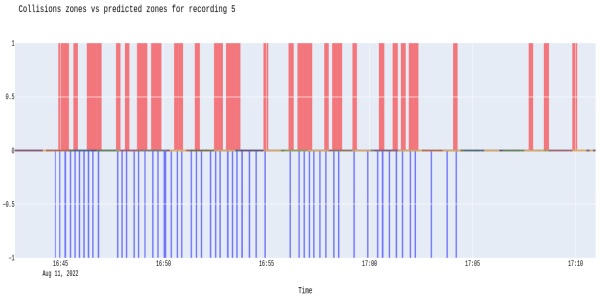

Fig. 26. Detected collisions (in red) vs labeled collisions (in blue) for LSTM-ED on recording 5.

V. CONCLUSIONS AND FUTURE WORKS

In this paper, we compared seven SOTA models belonging to different typologies against an ad-hoc method developed during classes. The results of the Bayesian model were good with respect the mean accuracy showed in tables, bringing a new method to analyse signals in this task; this could be useful for the upcoming research with action-based approaches. Furthermore, we found that some of the considered SOTA outperformed other existing models in the anomaly classification such as Random Forest and LSTM-ED, while other ones are not so good such as DAGMM. This can be seen by just looking at the ROC curves. LSTM-ED, on the other hand, leads to good results relying on its architecture that can learn the trend of a signal by the encoder-decoder structure, minimizing the reconstruction error. Instead, DAGMM proved itself to be a not suited architecture for our application. A possible cause could be the high amount of features of the dataset; due to this reason the model could not build a good gaussian approximating well enough the data that it receives. Overall we analyzed the proposed dataset with

different models and approaches opening the way for further research on the analysis of the Kuka robot context.

REFERENCES

[1] M. Liu, G. Liao, N. Zhao, H. Song, and F. Gong, "Data-driven deep learning for signal classification in industrial cognitive radio networks," *IEEE Transactions on Industrial Informatics*, vol. 17, pp. 3412–3421, 2021.

[2] S. Krishnan and Y. Athavale, "Trends in biomedical signal feature extraction," *Biomed. Signal Process. Control.*, vol. 43, pp. 41–63, 2018.

[3] M. Canizo, I. Triguero, A. Conde, and E. Onieva, "Multi-head cnn-rnn for multi-time series anomaly detection: An industrial case study," *Neurocomputing*, vol. 363, pp. 246–260, 2019.

[4] Z. Hajiabotorabi, A. Kazemi, F. Samavati, and F. M. M. Ghaini, "Improving dwt-rnn model via b-spline wavelet multiresolution to forecast a high-frequency time series," *Expert Syst. Appl.*, vol. 138, 2019.

[5] M. Wiese, R. Knobloch, R. Korn, and P. Kretschmer, "Quant gans: deep generation of financial time series," *Quantitative Finance*, vol. 20, pp. 1419 – 1440, 2019.

[6] M. A. Bashar and R. Nayak, "Tanogan: Time series anomaly detection with generative adversarial networks," in *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, Dec. 2020. [Online]. Available: http://dx.doi.org/10.1109/SSCI47803.2020.9308512

[7] A. Y. Yildiz, E. Koç, and A. Koç, "Multivariate time series imputation with transformers," *IEEE Signal Processing Letters*, vol. 29, pp. 2517–2521, 2022.

[8] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16. ACM, Aug. 2016. [Online]. Available: http://dx.doi.org/10.1145/2939672.2939785

[9] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *2008 Eighth IEEE International Conference on Data Mining*, 2008, pp. 413–422.

[10] J. Li, H. Izakian, W. Pedrycz, and I. Jamal, "Clustering-based anomaly detection in multivariate time series data," *Appl. Soft Comput.*, vol. 100, p. 106919, 2021.

[11] Q. Ma, S. Li, W. Zhuang, J. Wang, and D. Zeng, "Self-supervised time series clustering with model-based dynamics," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, pp. 3942–3955, 2020.

[12] L. Breiman, "Multivariate time series imputation with transformers," *Machine Learning*, vol. 45, p. 5–32, 10 2001.

[13] S. Liu, B. Zhou, Q.-X. Ding, B. Hooi, Z. Zhang, H. Shen, and X. Cheng, "Time series anomaly detection with adversarial reconstruction networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, pp. 4293–4306, 2023.

[14] G. Li and J. Jung, "Deep learning for anomaly detection in multivariate time series: Approaches, applications, and challenges," *Information Fusion*, vol. 91, 10 2022.

[15] J. Schneider, P. Wenig, and T. Papenbrock, "Distributed detection of sequential anomalies in univariate time series," *The VLDB Journal*, vol. 30, pp. 579 – 602, 2021.

[16] S. Tuli, G. Casale, and N. Jennings, "Tranad: Deep transformer networks for anomaly detection in multivariate time series data," *Proc. VLDB Endow.*, vol. 15, pp. 1201–1214, 2022.

[17] S. Kim, K. Choi, H.-S. Choi, B. Lee, and S. Yoon, "Towards a rigorous evaluation of time-series anomaly detection," 2022. [Online]. Available: https://arxiv.org/abs/2109.05257

[18] S. Schmidl, P. Wenig, and T. Papenbrock, "Anomaly detection in time series: A comprehensive evaluation," *Proc. VLDB Endow.*, vol. 15, pp. 1779–1797, 2022.

[19] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16. ACM, Aug. 2016. [Online]. Available: http://dx.doi.org/10.1145/2939672.2939785

[20] C. Leys, C. Ley, O. Klein, P. Bernard, and L. Licata, "Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median," *Journal of Experimental Social Psychology*, vol. 49, pp. 764–766, 2013.