ANOMALY DETECTION ON TIME-SERIES DATA IN ENERGY SECTOR

Chiradip Biswas
3rd Year Undergraduate
Indian Institute of Technology Kharagpur
Kharagpur, India
chiradipb02@kgpian.iitkgp.ac.in

Devodita Chakravarty
3rd Year Undergraduate
Indian Institute of Technology Kharagpur
Kharagpur, India
devoditac@kgpian.iitkgp.ac.in

Ishani Ghosh
3rd Year Undergraduate
Indian Institute of Technology Kharagpur
Kharagpur, India
ishanighosh@kgpian.iitkgp.ac.in

Biswajeet Nayek
3rd Year Undergraduate
Indian Institute of Technology Kharagpur
Kharagpur, India
nayakbiswajeet04@kgpian.iitkgp.ac.in

ABSTRACT

Energy systems such as electric grids and power systems in buildings, wind turbines and spacecrafts for extra-terrestrial exploration play an important role in industrial and research domains. However, monitoring of system health is often neglected or inefficiently carried out in these industrial processes that leads to large amounts of energy wastage, machine damages and safety hazards. Approximately 20% of the valuable energy that mankind is trying to conserve for sustainability gets wasted due to negligence or delayed response to equipment failure or misconfiguration in devices. Most energy systems consist of multiple devices integrated into a single complex unit, making it challenging to monitor individual components. Consequently, detecting malfunctions relies on identifying anomalous patterns in their time series outputs. In this paper, we explore two modern methods, TadGAN and AER for efficient time series anomaly detection in the energy sphere. We attempt to highlight the unique architecture of these models distinguishing them from other deep learning and statistical techniques, and explore their performance on several datasets, after pre-processing and exploratory data analysis. We propose the use of evaluation metrics such as ROC-AUC, PR curve and contextual F1, accuracy scores and report the best outcomes, after hyperparameter tuning. We have also explored techniques to improve model performance on some datasets, after identifying the points where they failed. Finally, we compare both the models and analyze their limitations, especially from an industrial perspective and discuss possible future enhancements to improve them.

***K*eywords** Time Series Analysis · Anomaly Detection · GANs · Auto-encoders

1 Introduction

Anomaly detection is the identification of rare events, items, or observations which are suspicious because they differ significantly from standard behaviors or patterns. Human eye is an impressive tool for this task but, the anomalies are not always visible from the common inspection to the data. So we use various ML algorithms to detect them by analyzing the data and their patterns. Such anomalies are also seen in time-series data. A time series anomaly is defined as a time-point or period where a system behaves unusually.

Time series anomaly detection is important for a wide range of research fields and applications, including financial markets, economics, earth sciences, manufacturing, and healthcare. The presence of anomalies can indicate novel or unexpected events, such as production faults, system defects, and heart palpitations, and is therefore of particular interest.

Broadly speaking, there are generally 2 classes of these anomalies:

1. **Point-anomaly :** A single data point that has reached an unusual value.
2. **Collective-anomaly :** A continuous sequence of data points that are considered anomalous as a whole, even if the individual data points may not be unusual.

There is another classification of anomalies which is more detailed.

Anomalies in time-series can be classified as **temporal**, **inter-metric** , or **temporal-inter-metric** anomalies [2]. In a time series, temporal anomalies can be compared with either their neighbors (local) or the whole time series (global), and they present different forms depending on their behavior.

Different types of temporal anomalies are as follows:

- **Global**: These are spikes in the series, which are points with extreme values compared to the rest of the series. A global anomaly, for example, is an unusually large payment by a customer on a typical day. Considering a threshold, it can be described via the Eq. (1).

$$|x_t - \hat{x}_t| > \text{threshold} \tag{1}$$

  where, $\hat{x}$ is the output of the model. If the difference between the output and actual point value is greater than a threshold, then it is recognized as an anomaly.

- **Contextual**: A deviation from a given context is defined as a deviation from a neighboring time point, specifically one that lies within a certain range of proximity. These types of anomalies are small glitches in sequential data, where values deviate from their neighbors. A point can be normal in one context while being an anomaly in another. For example, large interactions, such as those on Boxing Day, are considered normal, but not on other days. The formula is the same as for a global anomaly, but the threshold for identifying anomalies differs. This threshold is determined by considering the context of neighboring points, as shown in Eq. (2).

$$\text{threshold} \approx \lambda \times \text{var}(X_{(t-w:t)}) \tag{2}$$

  here, $X_(t - w : t)$ refers to the context of the data point x_t with a window size *w*, var is the variance of the context of data point and λ controlling coefficient for the threshold. The second blue highlight in the figure given below is a contextual anomaly that occurs locally in a specific context.

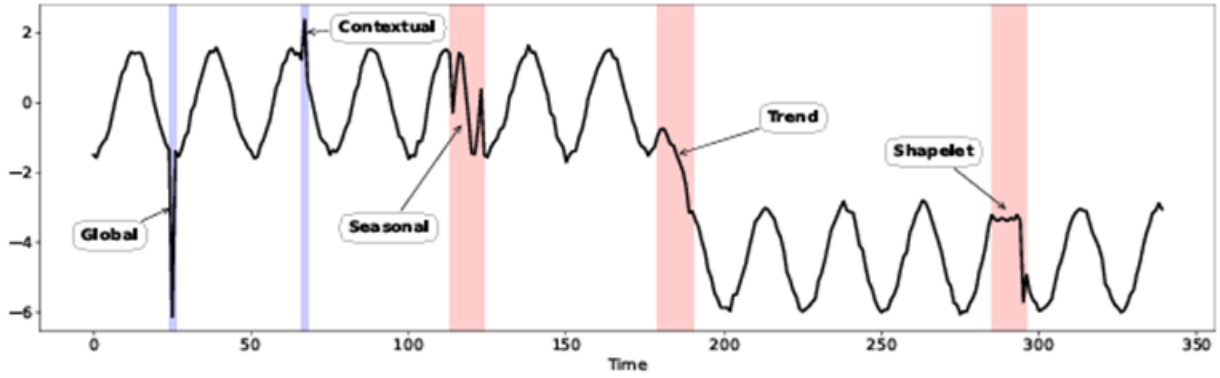


Figure 1: Time series data with various kinds of anomalous regions [2]

- **Seasonal**: Despite the normal shapes and trends of the time series, the seasonality can be unusual compared to the overall pattern. For example, the number of customers in a restaurant typically shows a clear weekly seasonality. Therefore, it's logical to identify deviations in this seasonality and address the anomalous periods individually.

$$diss(S, \hat{S}) > \text{threshold} \tag{3}$$

  In Eq. (3) *diss* is a function measuring the dissimilarity between two sub-sequences and $\hat{S}$ denotes the seasonality of the expected sub-sequence.

- **Trend**: A trend anomaly is an event that causes a permanent shift in the data's mean, leading to a transition in the trend of the time series. While the cycle and seasonality remain normal, the slope changes drastically.

Trends can occasionally change direction, switching from increasing to decreasing, or vice versa. For example, when a new song is released, it becomes popular for a while and then disappears from the charts, as shown in the segment in Fig. 1 where the trend changes and is assumed to be a trend anomaly. It is likely that the trend will restart in the future.

$$disss(T, \hat{T}) > \text{threshold} \tag{4}$$

In Eq. (4) $\hat{T}$ is the trend of the expected sub-sequence, and T is the normal trend.

- **Shapelet** : A shapelet is a distinctive time series sub-sequence pattern. This sub-sequence has a time series pattern or cycle that differs from the usual pattern found in the rest of the sequence. Variations in economic conditions, such as the total demand for and supply of goods and services, often cause these fluctuations. In the short run, these changes lead to periods of expansion and recession. Eq. (5) shows a mathematical representation of shapelet.

$$disss(C, \hat{C}) > \text{threshold} \tag{5}$$

where $\hat{C}$ specifies the cycle or shape of expected sub-sequences.

Measure of the Dissimilarity : These can be characterized by the distance between the actual sub-sequence observed and the expected sub-sequence. In this context, dynamic time warping (DTW), which optimally aligns two time series, is a great method for measuring this dissimilarity.

1.1 Problems Faced

The anomaly detection is generally an unsupervised classification problem, where the number of anomalous instances is very low compared to the normal instances. The datasets should contain labels in order to validate the models and for this, these datasets need to be examined by domain experts, who can mark the anomalous instances.

1. The main problem faced in the process was to find these labelled time series data, especially in the domain of energy. Mostly, even if the time-series data is available, the labels were missing. So the performance could be evaluated in labelled data only. On unlabeled data, we had to rely upon visually finding extent of detection of anomalous sequences, and the quality of reconstruction. We used the model's pipeline function for detecting the anomalous regions, in this case.
2. The models used in this process are deep learning based complex models[GAN, auto-encoder]. So training them is time-consuming, especially the TadGAN.

But even after these problems, we tried to find the best results possible for the models in each dataset by hyper-parameter tuning and also provided the basic statistical observations of the scores of models (mean and standard deviation).

1.2 Objectives

1. In this project, we tried to look into the performance of some of the existing methods for time-series anomaly detection. We tested and benchmarked the following models.
   - TadGAN [3] (Time-Series anomaly detection GAN)
   - AER [8] (Autoencoder Regression).
2. Reproducing the paper results.
3. Using the models on the datasets mentioned in the research papers and finding AUC-ROC scores on them, to understand performance of the model independent of threshold.
4. Benchmarking the models on Energy sector, time-series datasets, on the basis of AUC-ROC, average precision score. We also found F1 score, recall, precision [using the optimal threshold from AUC-ROC curve using Youden J statistic and using that threshold to find the metrics in the way described in corresponding papers.]
5. Discussion of the performance of these 2 models on each dataset.
6. Ways to improve performance
7. Advantages and disadvantages of the models.

   The notebook, other library sources used are given in footnotes [1]

[1]The reference materials for this project are available here

2 Literature Review

2.1 Classes of Existing Methods

1. **Proximity-based methods**
   (a) Proximity-based methods detect anomalies by measuring the distance between objects, identifying those that are distant as anomalies.
   (b) These methods can focus on individual data points for point anomalies or sequences of data points for collective anomalies.
   (c) Within this category, distance-based methods like K-Nearest Neighbor (KNN) use a specified radius to define neighbors and determine an anomaly score based on the number of neighbors, while density-based methods such as Local Outlier Factor (LOF) and Clustering-Based Local Outlier Factor consider the density of an object and its neighbors.
   (d) However, these methods have significant limitations when applied to time series data, as they require prior knowledge of the duration and number of anomalies and are unable to capture temporal correlations.
2. **Prediction-Based Methods**
   (a) Prediction-based methods develop a model to predict future values of a time series and identify anomalies when the difference between the predicted and actual values exceeds a threshold.
   (b) Traditional statistical models like ARIMA, Holt-Winters, and FDA can be used but are sensitive to parameter selection and require significant domain knowledge and assumptions.
   (c) To address these issues, machine learning approaches have been developed. For example, Hierarchical Temporal Memory (HTM) is an unsupervised online sequence memory algorithm that detects anomalies in streaming data by encoding the current input into a hidden state and predicting the next state, with anomalies flagged based on prediction errors.
   (d) Similarly, Hundman et al. introduced Long Short Term Memory Recurrent Neural Networks (LSTM RNNs) to predict future time steps and detect anomalies by identifying large deviations from these predictions.
3. **Reconstruction-based methods**
   (a) Reconstruction-based methods develop a model to capture the latent structure (low-dimensional representations) of the given time series data and generate a synthetic reconstruction.
   (b) These methods assume that anomalies lose information when mapped to a lower-dimensional space and thus cannot be effectively reconstructed.
   (c) Consequently, high reconstruction errors indicate a high likelihood of anomalies.

2.2 Some of the Existing Methods

1. **LSTM-based Encoder-Decoder for Multi-sensor Anomaly Detection** [5]

   (a) **Overview**
   The LSTM-based Encoder-Decoder model for anomaly detection, referred to as EncDec-AD, works by learning to reconstruct 'normal' time-series data from multiple sensors. An LSTM encoder maps the input sequence to a fixed-dimensional vector representation, while an LSTM decoder reconstructs the sequence from this representation. The model is trained on normal sequences, so when it encounters an anomalous sequence, the reconstruction error will be significantly higher due to its inability to accurately reproduce the abnormal patterns. This reconstruction error is used to compute an anomaly score, with higher scores indicating a higher likelihood of anomalies. This method is robust across various types of time-series data, including predictable, unpredictable, periodic, aperiodic, and quasi-periodic sequences.

   (b) **Pros**

   - **Versatile**: Effective for a wide range of time-series types (predictable, unpredictable, periodic, aperiodic, quasi-periodic).
   - **Anomaly Detection without Anomalous Data**: Can be trained solely on normal data, useful when anomalous data is scarce.
   - **Robust Performance**: Demonstrates significant capability in detecting anomalies in both short and long time-series.

- **High Positive Likelihood Ratios**: Provides significantly higher anomaly scores for anomalous points compared to normal points, enhancing detection reliability.
- **Adaptability**: Able to work with multi-sensor data, improving its applicability in various real-world scenarios.

(c) **Cons**

- **Training Complexity**: Requires significant computational resources for training, especially with large datasets.
- **Parameter Sensitivity**: Performance can be sensitive to the choice of parameters and model architecture.
- **Interpretability**: The model's decisions can be difficult to interpret due to the complexity of LSTM networks.

(d) **Best Dataset Types**

- Works best on datasets with a mix of predictable and unpredictable behaviors.
- Effective on both periodic and aperiodic time-series data.
- Suitable for multi-sensor datasets where different sensors capture varied aspects of the system's behavior.

(e) **Types of Anomalies Detected**

- Can identify anomalies in both predictable and unpredictable time-series data.
- Detects anomalies in short sequences (as small as 30) and long sequences (as large as 500).
- Effective at finding anomalies in periodic, aperiodic, and quasi-periodic data.

2. **MTAD-GAT (Multivariate Time-series Anomaly Detection via Graph Attention Network)** [10]

(a) **Overview**

The MTAD-GAT model addresses multivariate time-series anomaly detection by leveraging two parallel graph attention layers: the feature-oriented and the time-oriented graph attention layers. The feature-oriented layer captures causal relationships between different features, treating each univariate time-series as a node in a complete graph. The time-oriented layer captures temporal dependencies by considering timestamps within a sliding window as nodes in a graph. These attention layers are combined with a Gated Recurrent Unit (GRU) layer to capture long-term dependencies in the data. The model jointly optimizes a forecasting-based model, which focuses on single-timestamp prediction, and a reconstruction-based model, which captures the entire time-series distribution. This joint optimization allows the model to learn better representations of the time-series data, enhancing anomaly detection performance.

(b) **Pros**

- **Captures Multivariate Correlations:** Effectively models relationships between different time-series features without prior knowledge.
- **Temporal Dependencies:** Efficiently captures temporal dependencies using the time-oriented graph attention layer.
- **Joint Optimization:** Combines the strengths of forecasting and reconstruction models for improved accuracy.
- **High Performance:** Demonstrates superior performance on various datasets, achieving state-of-the-art results.
- **Good Interpretability:** Provides interpretable results by analyzing attention scores, which correspond well to human intuition.
- **Robust Preprocessing:** Includes normalization and data cleaning to enhance model robustness.

(c) **Cons**

- **Complexity:** The model's complexity might result in longer training and inference times.
- **Parameter Sensitivity:** Performance may depend on careful tuning of model parameters.
- **Data Requirement:** Requires a large amount of data for training to achieve optimal performance.
- **Overfitting Risk:** There is a potential risk of overfitting, especially with noisy data or small datasets.

(d) **Best dataset Type**

The MTAD-GAT model works best on datasets with multiple correlated time-series features where capturing the interplay between different features and their temporal dynamics is crucial.

(e) **Best at Identifying**

MTAD-GAT is particularly effective at identifying contextual and collective anomalies, where the relationships between different time-series and their temporal patterns play a significant role in defining normal and anomalous behavior.

3. **A Deep Neural Network for Unsupervised Anomaly Detection and Diagnosis in Multivariate Time Series Data** [9]

(a) **Overview**

The Deep Neural Network for Unsupervised Anomaly Detection and Diagnosis in Multivariate Time Series Data, known as MSCRED, operates by constructing multi-scale signature matrices to capture system statuses at different levels. It utilizes a convolutional encoder to encode inter-sensor correlations and a ConvLSTM network to capture temporal patterns. The model then reconstructs signature matrices and uses a square loss for end-to-end learning, enabling effective anomaly detection and diagnosis in multivariate time series data.

(b) **Pros**

- **Effective Anomaly Detection**: MSCRED outperforms baseline methods, showing superior performance in detecting anomalies.
- **Root Cause Identification**: It can accurately identify the root causes of anomalies, aiding in pinpointing the source of issues.
- **Anomaly Severity Interpretation**: The model can interpret the severity of anomalies, distinguishing between short, medium, and long-duration anomalies.
- **Robustness to Noise**: MSCRED demonstrates robustness to input noise, enhancing its reliability in real-world applications.

(c) **Cons**

- **Complexity**: The model may be complex due to its multi-scale approach and neural network architecture.
- **Training Data Dependency**: Performance may vary based on the quality and quantity of training data.
- **Interpretability**: Interpretation of the model's decisions may be challenging due to the complexity of deep learning algorithms.

(d) **Best Dataset and Anomaly Types**

MSCRED works best on datasets with multivariate time series data, such as power plant data, where correlations between different pairs of time series are crucial. It excels at identifying anomalies of varying durations, including short, medium, and long-duration anomalies, making it suitable for detecting a wide range of anomaly types in complex systems.

2.3 Motivation for choosing TadGAN [3]

Effective time series anomaly detection involves capturing underlying temporal correlations in data sequences, and then isolating points which deviate from these patterns. These sequential dependencies may be non-linear, and hence difficult to analyze with conventional statistical tools. Deep learning serves as a boon in capturing these complex relations, and has excellent learning capability of recurrent non-linear patterns. Several models learn the underlying relations and predict or reconstruct the original signal from a latent mapping of features, and then make a comparison between the real and reconstructed values. A high reconstruction or prediction error is then used to flag anomalies.

However, one of the fundamental drawbacks of these methods in time series detection lies in overfitting to the time series such that even anomalous points get reconstructed

Deep learning methods' like autoencoders use L2 norms as part of their loss functions to judge similarity between the real and reconstructed signal, which makes them so accurate that deviations and anomalies are also faithfully reproduced. Presence of anomalous points in the reconstructed signal results in a small reconstruction error at these points, which prevent anomalies from being flagged, leading to false negatives.

TadGAN (Time Series Anomaly Detection using Generative Adversarial Neural Networks) is a GAN novel architecture which can effectively capture and reconstruct temporal dependencies in the data without overfitting to anomalous regions.

A generative adversarial neural network consists of two networks training in competition to one another- the Generator which generates data points as an attempt to construct fake instances from white noise, and the Discriminator which acts as a critic and distinguishes fake instances from the corresponding real ones. With every training iteration, the Generator and Discriminator compete against each other to minimize their adversarial losses. The Generator does so by effectively fooling the Discriminator by producing more real instances which match the learned patterns and underlying distribution of the data. If the Discriminator correctly identifies fake data, the generator receives feedback and tries to better resemble the data by capturing more relations and patterns, while if the discriminator misclassifies real and fake instances, it is trained to sharpen its ability to distinguish real and fake data.

However, several problems lie with existing GAN models, one of them being the phenomenon of "mode collapse" in standard adversarial loss during the training phrase, where the Generator finds a limited variety of inputs that can consistently fool the discriminator, and prefers to keep on producing those, with a reluctance to explore the complete range of the data distribution.

TadGAN mitigates this problem of mode collapse as well as the overfitting of time series challenges, with its novel architecture. The model consists of two Generators, one which acts as an encoder and is responsible for mapping the signal into a latent space and another which acts as a decoder that converts the latent encoded features into a reconstruction of the original time series. Additionally, there are two adversarial Critics, the critic Cx which is responsible for distinguishing real signals from fake ones, and the critic Cz distinguishes the latent encoding from random white noise samples. The critic Cx is trained on Wasserstein loss which tackles the problem of mode collapse and cycle consistency loss where the signal is encoded and then decoded, and the difference between the reconstruction and original signal is minimized with an L2 norm. This further reduces mapping to the search space and helps in better capturing of temporal dependencies. One of the most striking features of this model is a second discriminator Cz which distinguishes the latent encoding from white noise samples. This helps in regularizing the encoder and prevents it from overfitting into the anomalous regions, in contrast to other models. Any contradiction between the encoder E and generator (decoder-G) is prevented by the cycle consistency loss.

The error function(between real and constructed signal) is created by combining the reconstruction error as well as the critic score. This is because training will make the critic highly sensitive to the underlying pattern of the data, and any deviations from its learning, in the form of anomalies will be flagged as fake, in the form of a low critic score. Thus a low critic score combined with a high reconstruction error strongly indicates anomalous regions and the final function is a weighted combination of both.

Thus, this regularized mapping to a lower dimensional space followed by a reconstruction capturing temporal correlations without reproducing anomalies, and the innovative use of critic score of a GAN combined with reconstruction errors to flag positives indicates immense potential in anomaly detection, inspiring us to explore its performance and behavior on various datasets.

2.4 Motivation for choosing AER [8]

Traditionally, anomaly detection has been approached using one of two primary methodologies: reconstruction-based methods, such as Long Short-Term Memory Autoencoders (LSTM-AE) and Time-series Anomaly Detection using

Generative Adversarial Networks (TadGAN), and prediction-based methods, such as AutoRegressive Integrated Moving Average (ARIMA) models and Long Short-Term Memory networks combined with Decision Trees (LSTM-DT). Each of these methodologies has distinct advantages and disadvantages, which tend to complement each other. Reconstruction-based methods are particularly effective in identifying contextual anomalies, which are groups of data points that fall within the series' normal range but do not follow expected temporal patterns. However, these methods often struggle to identify point-based anomalies, which are single data points that deviate significantly from the rest of the data. On the other hand, prediction-based methods excel at detecting point anomalies but often have difficulty with contextual anomalies. Additionally, these methods can produce false positives and miss predictions at early indices.

In the context of anomaly detection, a novel approach called AER (Auto-Encoder with Regression) seeks to integrate the strengths of both prediction and reconstruction-based methods while addressing some of their inherent challenges. Although the MTAD-GAT model also combines these concepts, AER differentiates itself by utilizing bidirectional LSTM (BiLSTM) networks instead of the graph-based methods used in MTAD-GAT, which require extensive pre-processing. BiLSTM is particularly advantageous for capturing temporal dependencies in sequential data, making it well-suited for our application.

The AER method builds on the foundations of LSTM-DT and LSTM-AE. It employs a conventional encoder, but its decoder reconstructs the sequence at time step of two more than the encoded, incorporating three components: one-step reverse prediction, the reconstructed sequence, and one-step ahead prediction. This design helps to address false positives generated by exponential weighted moving average smoothing functions by masking certain indices at the start of the sequence with a minimum anomaly score value. This approach mitigates the need for the first few observations to produce the initial forecast.

Anomaly scores in AER are calculated in two directions: forward and reverse. The first few values of the forward predictions and the last few values of the backward predictions are padded with zeros. For non-overlapping segments of the sequence, the maximum value is taken, while for overlapping segments, the average of both directions is computed. The prediction loss is defined as the average mean squared error between pairs of true and predicted values in both reverse and forward directions. The reconstruction loss is calculated as the mean squared error between the original time series and the reconstructed sequence. The overall loss is a weighted combination of both prediction and reconstruction losses, determined by a parameter gamma.

To combine anomaly scores, AER employs two methods: a convex sum and a product method, both of which use a weight parameter beta to control the relative importance of each component. The scores are scaled to [0,1] for the convex method and [1,2] for the product method. Additionally, AER implements a locally adaptive thresholding function to identify anomalous intervals from the anomaly scores. This function uses a sliding window to compute local thresholds. Observations with scores exceeding this threshold are flagged as anomalous. Continuous anomalous observations are merged to form sequences, and false positives are mitigated by pruning sequences that do not exceed an empirically defined percentage change threshold theta. Subsequent sequences that fail to meet this threshold are reclassified as normal, thereby reducing the number of false positives and improving the overall robustness of the anomaly detection process.

These considerations and innovations are the reasons that motivated us to choose this method for our research investigation. By integrating the strengths of both prediction-based and reconstruction-based approaches and by addressing their respective weaknesses, AER presents a comprehensive solution for anomaly detection in time-series data. Its ability to effectively handle both point and contextual anomalies, along with its robust mechanism to mitigate false positives and missing early predictions, makes it ideal for our study. Additionally, the use of BiLSTM networks enhances the model's capability to capture temporal dependencies, further improving its accuracy and reliability. By benchmarking and comparing this method with TadGAN, we aim to establish a clear understanding of its performance and potential applications in real-world scenarios.

2.5 Impact on Research World

The AER (Anomaly Detection using Encoder-Recurrent Decoder) and TADGAN (Time-series Anomaly Detection using Generative Adversarial Networks) models can significantly impact various positive aspects of the world by enhancing anomaly detection capabilities across different domains. Here are some key impacts:

1. **Healthcare**:
    - **Early Detection of Diseases**: These models can identify anomalies in patient data, leading to early detection of diseases such as cancer or heart conditions, potentially saving lives.

- **Monitoring Patient Vital Signs**: Continuous monitoring of patient vitals can detect irregular patterns, allowing for timely medical interventions.

2. **Finance**:
   - **Fraud Detection**: By identifying unusual patterns in transaction data, these models can help detect and prevent fraudulent activities, protecting consumers and financial institutions.
   - **Risk Management**: Anomaly detection can alert to market irregularities, helping in better risk assessment and management.

3. **Industry and Manufacturing**:
   - **Predictive Maintenance**: Detecting anomalies in machinery performance data can predict equipment failures before they happen, reducing downtime and maintenance costs.
   - **Quality Control**: Identifying defects in production processes early ensures higher product quality and reduces waste.

4. **Cybersecurity**:
   - **Intrusion Detection**: These models can detect unusual network activity, helping to identify and mitigate cyber-attacks promptly.
   - **Data Breach Prevention**: By monitoring data flow, these models can detect and prevent potential data breaches, protecting sensitive information.

5. **Environmental Monitoring**:
   - **Climate Change and Pollution Tracking**: Detecting anomalies in environmental data can help monitor and respond to changes in climate patterns or pollution levels, contributing to environmental conservation efforts.
   - **Natural Disaster Prediction**: Analyzing patterns in geological and meteorological data can aid in predicting natural disasters, allowing for better preparedness and response.

3 Experimental Setup

The general pre-processing steps available in the model pipeline [as mentioned in the reference research paper] were used by default. These steps are:—

1. **Time-segment aggregation:** Aggregating the time-series values over a given interval of time. The aggregation method used is averaging.
2. **Min-max scaling:** Normalize the values in the scale of [-1,1]
3. **Imputation of Null values:** Used keras simple imputer to impute null values
4. **Creating Rolling window sequences:** The function creates an array of input sequences and an array of target sequences by rolling over the input sequence with a specified window.

Finally, these windows are put together in a 3D array with each row representing the window sequences, and the 3rd dimension representing channels or number of features for the series. The target sequence is a sampled version of the column, the values of which we will predict/reconstruct using the model.

We used the following datasets to evaluate the model performance.

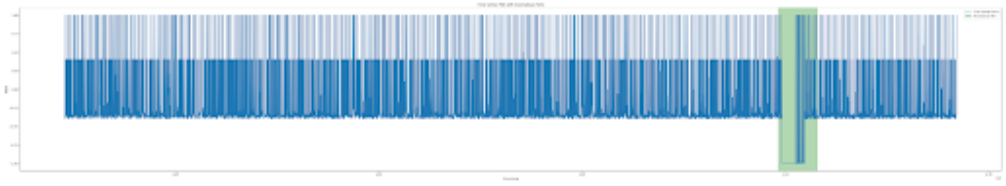
3.1 NASA space telemetry dataset

This dataset (as shown in Table 1) contains expert-labeled telemetry anomaly data from the Soil Moisture Active Passive (SMAP) satellite and the Mars Science Laboratory (MSL) rover, Curiosity. All telemetry channels discussed in an individual ISA were reviewed to ensure that the anomaly was evident in the associated telemetry data, and specific anomalous time ranges were manually labeled for each channel.
SMAP: TM Channels (55), Total TM values (429,735), Total anomalies (69),
MSL: TM Channels (27), Total TM values (66,709), Total anomalies (36)
Fig. 2 shows some of the sample signals with the anomalous regions.

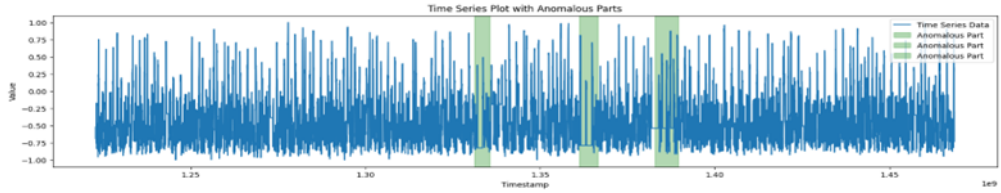
Table 1: NASA telemetry datasets used

Channel id	spacecraft	Anomalous sequences	class	Number of values
P-1	SMAP	[[2149, 2349]	contextual	8505
		[4536, 4844]	contextual	
		[3539, 3779]]	contextual	
S-1	SMAP	[[5300, 5747]]	point	7331
P-15	MSL	[[1390, 1410]]	point	2856

(a) P-15 channel signal from msl



(b) S-1 channel signal from smap



(c) P-1 channel signal from small (with contextual anomalies)

Figure 2: Some of the signals with the anomalous regions marked as green

3.1.1 Preprocessing steps taken for the NASA Telemetry data

As this data comes in the form of numpy arrays for each channel with the corresponding values. We convert them to CSV files with two columns: timestamp and value. For this, we perform loading both the train and test matrices for each signals and concatenate them to generate a single matrix for each signal. Thereafter, we add a timestamp column by taking the value 1222819200 (2008-10-01T00:00:00) as for the first row and then increasing the timestamp by 21600 seconds (6h) for each other row. Finally we store the data as a csv file, specific to the channel name, and the test and train sequences concatenated into a single time-series of that channel.

We used the 'labeled_anomalies.csv' file from the 'telemanom project' and converted it to the CSV. Later on we used the anomalous sequences from the orion library signals. The model results are given in the experimental results section.

3.2 SCADA 2015 Wind turbine Dataset

Supervisory Control and Data Acquisition (SCADA) systems are used for controlling, monitoring, and analyzing industrial devices and processes. The system consists of both software and hardware components and enables remote and on-site gathering of data from the industrial equipment. In that way, it allows companies to remotely manage industrial sites such as wind farms, because the company can access the turbine data and control them without

being on site. Using 10-minute wind turbine supervisory control and data acquisition (SCADA) system data to predict faults can be an attractive way of working toward a predictive maintenance strategy without needing to invest in extra hardware. Fig. 3 shows some of the sample signals with the anomalous regions. The dataset contains time series with the normalized measurement columns like : 'wind_speed', kw, wind_speed_sd, wind_speed_max, torque_actual_value, blade_1_actual_angle, blade_2_actual_angle, blade_3_actual_angle, blade_1_2_act_angle_diff, blade_2_3_act_angle_diff, blade_3_1_act_angle_diff. Besides, it also contains some counter columns, which are not normalized: 'ot', 'sot', 'dt', 'lot', 'wot', 'est', 'mt', 'rt', 'eect'.

3.2.1 Preprocessing steps taken for the data

The data was a time-series in 10 minute intervals. The intervals were continuous, except some cases, where the turbines stopped [these are considered anomalous, but if they meet certain conditions]. There were 2 csv files, scada_data.csv and events_data.csv .[The wtphm library was used to preprocess the events_data and label the scada_data]

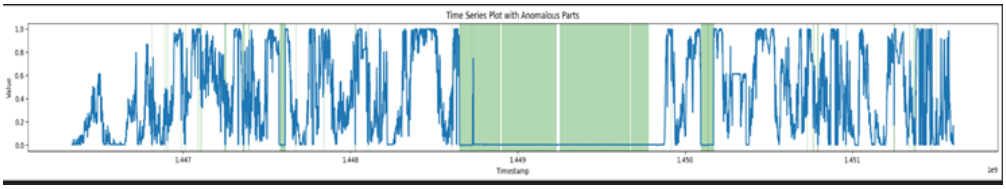
Processing the events data and making the anomalous regions: Regarding the stoppage of turbines, there was another dataset, the alarm data. All the time intervals, when turbine stops are enlisted there along with specific stop_codes [which we can use to identify which stoppages were due to faults in the turbine and which are not].

stop_cat: This is a category for events which cause the turbine to come to a stop. It could be the functional location of where in the turbine the event originated (e.g. pitch system), a category for grid-related events, that the turbine is down for testing or maintenance, in curtailment due to shadow flicker, etc. The "stop_cat" in the datasets were,

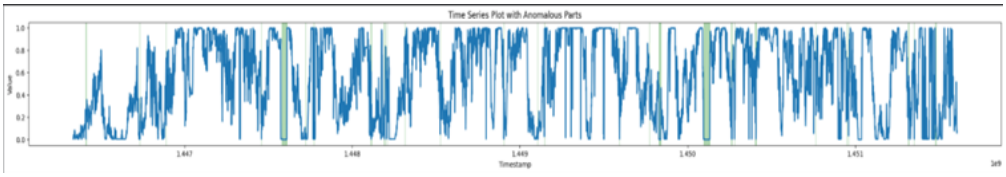
- OK—The turbine was operating normally.
- Down—The turbine was not operating due to a fault detected in one of its subsystems.
- Grid—The turbine was not operating due to a grid fault.
- Weath—The turbine was not operating, or was curtailed, because of severe weather.
- Maint—The turbine was down for routine/scheduled maintenance.
- Rep—The turbine was down for unplanned repairs.

We find the stop_codes for stop_cat like- fault, maintenance, grid,sensor, test. Then we group the events having faults, falling in the same class.

Batches were made of the consecutive time instances, with timestamp just after a stop_code and before ok_code. This also ensures to add the consecutive time intervals with the same class of fault into a continuous time interval. Now each batch is assigned the root cause to the stopping, based on stop_code distribution in a batch. Thus the stoppage intervals are labeled. Now we assign the root cause to all the time intervals [if consecutive, already merged] in events data. These act as the labels of each stoppage. Out of the labels, we consider anomalous intervals in which the labels contain 'fault', 'grid' or 'sensor'. Thus the anomalous instances were filtered out from the events data.



(a) Turbine-21



(b) Turbine-22

Figure 3: SCADA-2015 dataset with anomalous regions highlighted in green

Though the dataset contained no null values, some time-instances were missing from the data, especially when the turbine stopped. So we had to mitigate that issue in preprocessing separately, otherwise the default preprocessing of TadGAN or AER posed a problem.

We first added the missing rows, of the time instances falling in the interval. Then we imputed the values based on column meaning.

- kw: If the turbine is stopped, no power is generated. So impute with 0
- Torque_actual_value: When the turbine is stopped, no torque is moving the turbine. So fill 0.
- Blade angles: If the turbine does not move, the angles will be the same as before.
- Windspeed: This variable was filled with the value it attained before the absent row.
- Counter columns: The filling of these rows was left on the simple imputer of the models.

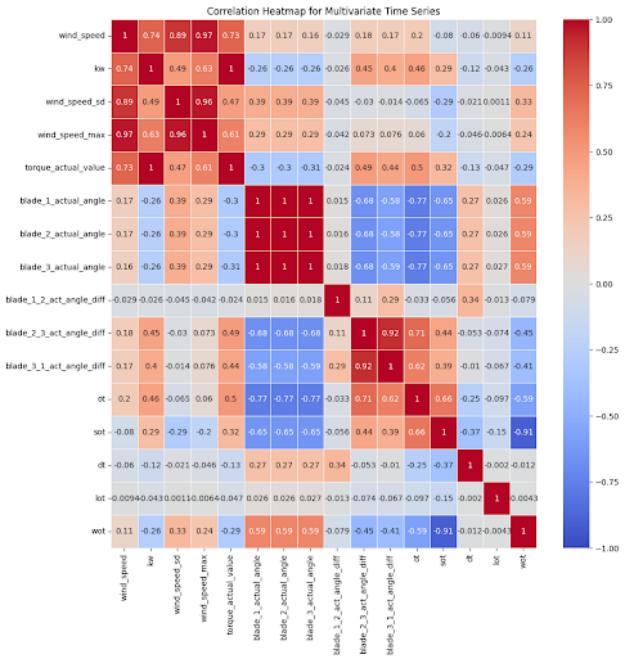


Figure 4: Correlation Heatmap for SCADA turbine-22 dataset

3.2.2 EDA on Scada Turbine 22 data

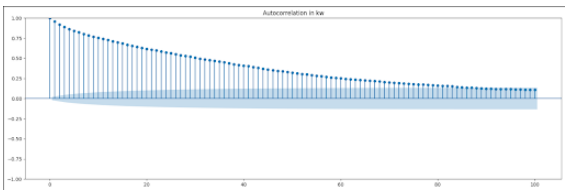
As the dataset is a multivariate time series, a correlation heatmap is used to find the correlation between the columns and feature selection. From Fig. 4, the below points are evident.

- 'kw' column is highly correlated with 'wind_speed' and 'actual torque value' columns.
- using wind_speed,does not need wind_speed max, std to be included
- windspeed and torque value also have high correlation, so either of them can be used
- But the counter columns have no correlation, so including them may give better results, capturing more features.

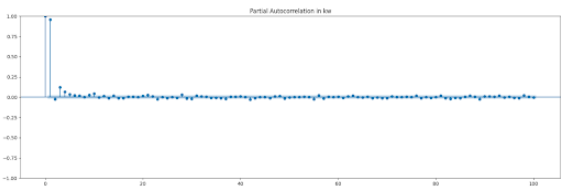
We next plot the Auto-correlation and Partial Auto-correlation plots, as shown in Fig. 5 to identify any periodic or trend nature of it. For Blade angle, we can observe a very weak periodicity (20) is observable initially, but it died down as lag increased.

From these plots, we can conclude that:

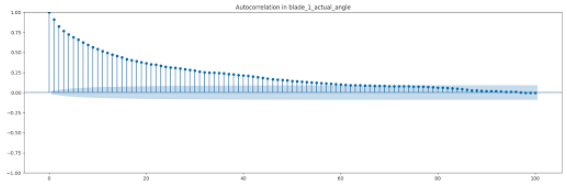
- Long correlation with previous values, long term trend, but weakens as lag increases. So there is a cumulative cascading effect. Only short term dependence is visible.
- There is no special period for seasonality. Even if it is present, it is very large, larger than the data length, as there are no spikes at regular intervals in the PACF plot.
- Slow monotonic decrease suggests a trend and non stationarity [very slight increasing]



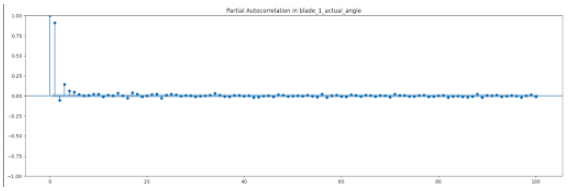
(a) Auto-correlation plot for the kw column



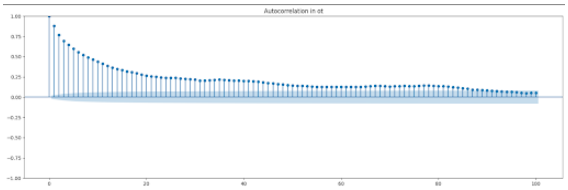
(b) Partial Auto-correlation plot for the kw column



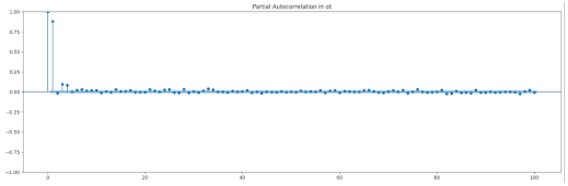
(c) Auto-correlation plot for blade 1 actual angle



(d) Partial auto-correlation plot for blade 1 actual angle



(e) Auto-correlation plot for the counter column 'ot'



(f) Partial auto-correlation plot for the counter column 'ot'

Figure 5: Auto-correlation and Partial Auto-correlation plots for different columns in the dataset

Overall, we observe that columns of all the classes have more or less the same time-series characteristics, short-term dependence, very weak monotonic trend, and no definite period.

3.3 Power Laws: Anomaly Detection Dataset

This dataset was given in a competition organized by "DRIVENDATA", to challenge people around the world, to make predictions on the electrical energy consumption of buildings and also detect unusual patterns. The data was made by 'Schneider Electric' and is available at Schneider Exchange. The dataset consisted of temperature variation and electric meter readings of several buildings from 2015-6-11 to 2017-9-19 , in 15 minute intervals. We apply the models on 2 particular meter ids (334_61,234_203).

The main problem in the dataset was lack of labels. But in the repository of the competition winners, they pointed out some of the abnormal regions. The anomalousness of these regions were also visually understood. Some were pointed out by the competition winners in their repositories, and we identified a few based upon some observations. For example:

1. In meter id 234_203 the reading is made of 2 separate meter readings, meter 863 (till 2015/6/20) and meter 938 (after that date). So there is a sudden jump in the series, which is in turn anomalous [sudden change of trend]. Also in the 2nd portion, at a certain timestamp, the reading was unusually high, globally high for the dataset. So it can be considered as temporal global point anomaly. From Fig. 6 we can observe, at locations 1,2,3,4,6 there is a sudden dip, but it occurs every year, so they may not be overall anomalous, but the data being very long, they may become contextual anomalies. Region 5 is the global point anomaly.
2. In meter 334_61 about 1 month readings are missing from 2017/3/17 to 2017/4/26. It may occur that at that time period the meter stopped working or connection was cut. On imputation [sklearn simple imputer or filling with 0] can be reasonable. So that portion can be considered anomalous. Besides there are portions where the readings dropped unusually for small duration and again followed the usual trend. So that may also be considered an anomalous region. Fig. 7 shows meter 334_61 with the anomalous regions 1,2 and 3 (the sudden increase of power consumption).
3. The meter 38_9687 readings were demand power readings and meter 38_9688 were corresponding reactive power readings. In Fig 8, we see that the trend of the series changed suddenly (after the red line); from a linearly increasing trend, it became almost constant. That portion can be considered anomalous.

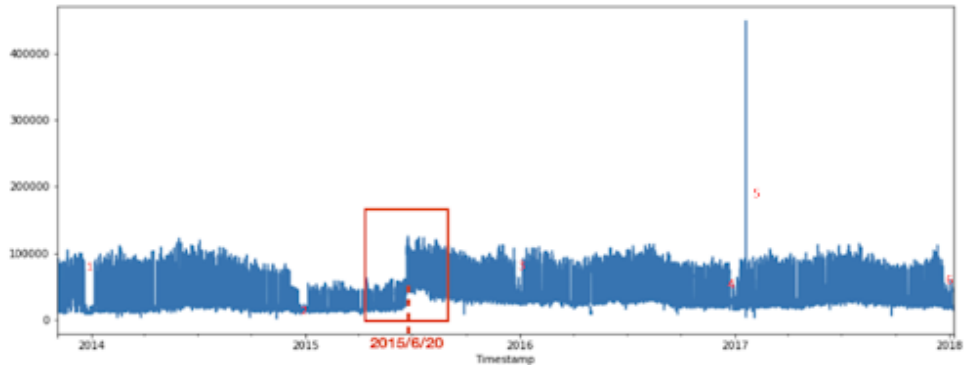


Figure 6: Reading of meter id 234_203

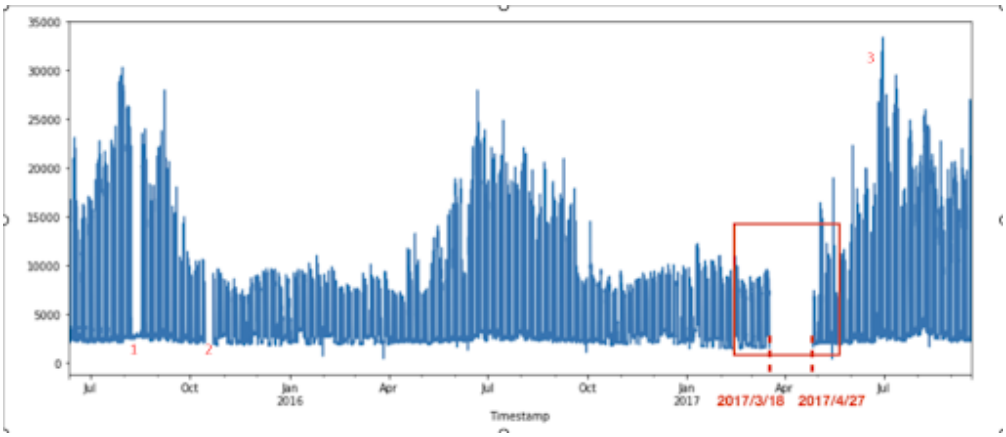


Figure 7: Reading of meter 334_61

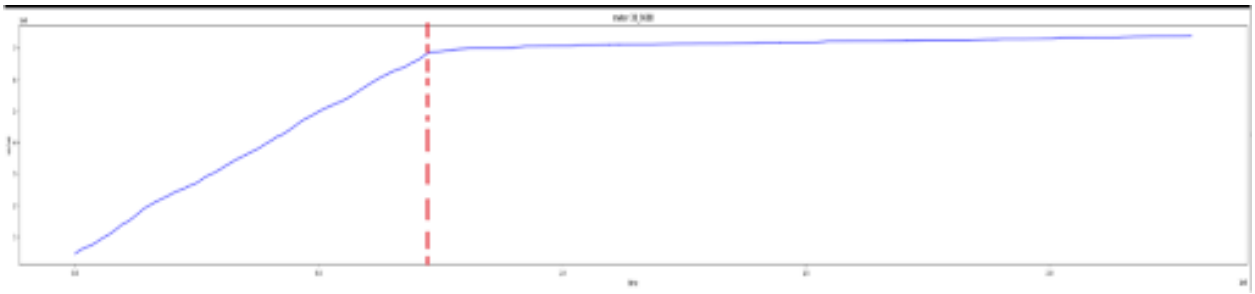


Figure 8: meter 36_9688 reactive power reading

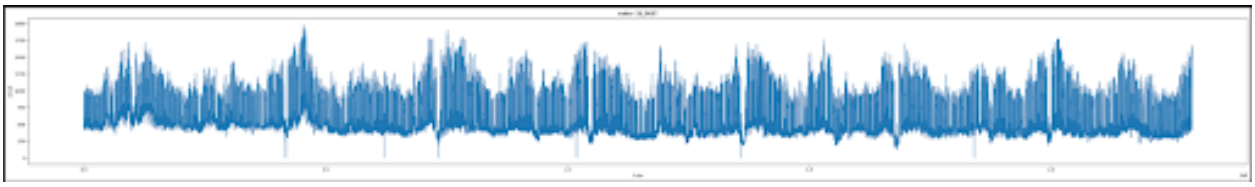


Figure 9: meter 36_9688 power demand readings

3.4 Machine Temperature Failure Dataset

It is a dataset, available in "Numenta Anomaly Benchmark" 's Git Repository, in the 'RealKnownCause' section. This data depicts the temperature of a machine in 5 minute intervals from (2013-12-02 21:15:00) till (2014-02-19 15:25:00).

It is a univariate time series data with 22695 instances and 2 columns (timestamp, value). It is a labeled dataset, and the anomalous instances are given in the form of points [particular time instances, not range].

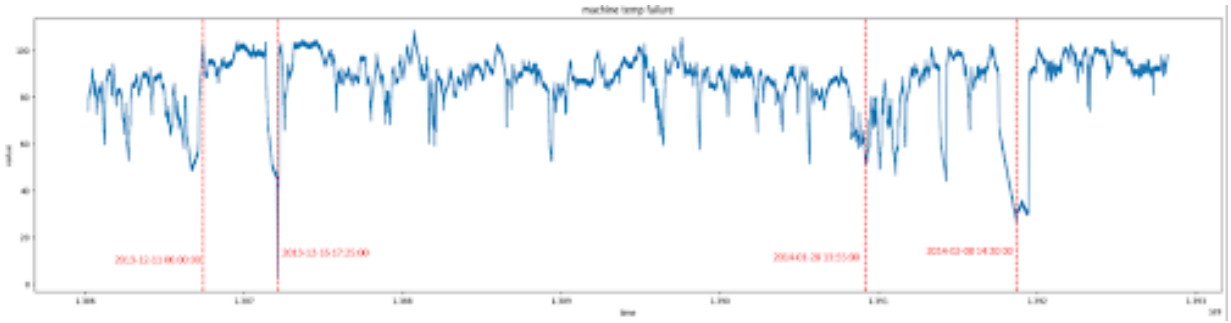


Figure 10: Machine Temperature failure dataset with labeled anomalous instance with red

The anomalous intervals were made by extending the anomalous portion about the red dotted lines, on both sides, by a certain amount [reference taken from the Tulog notebook where NYC dataset anomalies were made from similar lines].

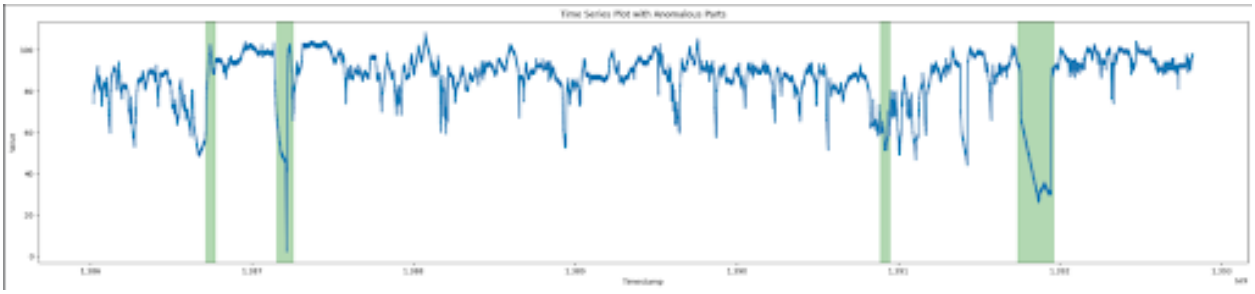


Figure 11: The green regions are the anomalous regions made from the lines

3.5 Large-scale Energy Anomaly Detection(LEAD)

This dataset was provided in a contest by Kaggle. It is an extended, labeled version of the dataset used the Great Energy Predictor III competition by ASHRAE(The American Society of Heating, Refrigerating and Air-Conditioning Engineers), where the makers have now annotated each datapoint of electric meters in 200 commercial buildings as being anomalous or non-anomalous. The purpose of this dataset is to leverage the thousands of time series values, obtained from energy meters for effective anomaly detection using analytical techniques and then use the labels to judge their efficacy. This can help quickly and automatically detect faulty behavior of the appliances and ensure immediate action, lowering the current 20% wastage of energy (used by buildings), due to malfunction of equipment, aging, incorrect configurations and human mishandling. It could potentially save valuable energy and be a significant stride towards achieving global sustainability.

3.5.1 Dataset Description

The dataset consists of electric meter readings, recorded every 1 hour, for a year, of 200 buildings. It is provided as a .csv file with 4 columns. The first column consists of the ‘building id’, unique for each of the 200 buildings and labeled as ‘1’, ’2’....’200’. The second column consists of the ‘time-stamp’ in date-time format. The third column has the numerical values of the ‘meter-readings’ recorded while the fourth column, ’anomaly’ contains the corresponding labels for each point, with ‘0’ marking them as non-anomalous and ‘1’ marking the point as anomalous.

3.5.2 Preprocessing Steps taken for the Data

The data being in .csv format was read into a dataframe by pandas. By filtering out building-wise time series data from the data, we noticed that several buildings had very few or no values, while others had large stretches of continuous values followed by chunks of NaN/null values. We filtered out 3 buildings with significant numbers of non-null values-buildings 107,108,139.

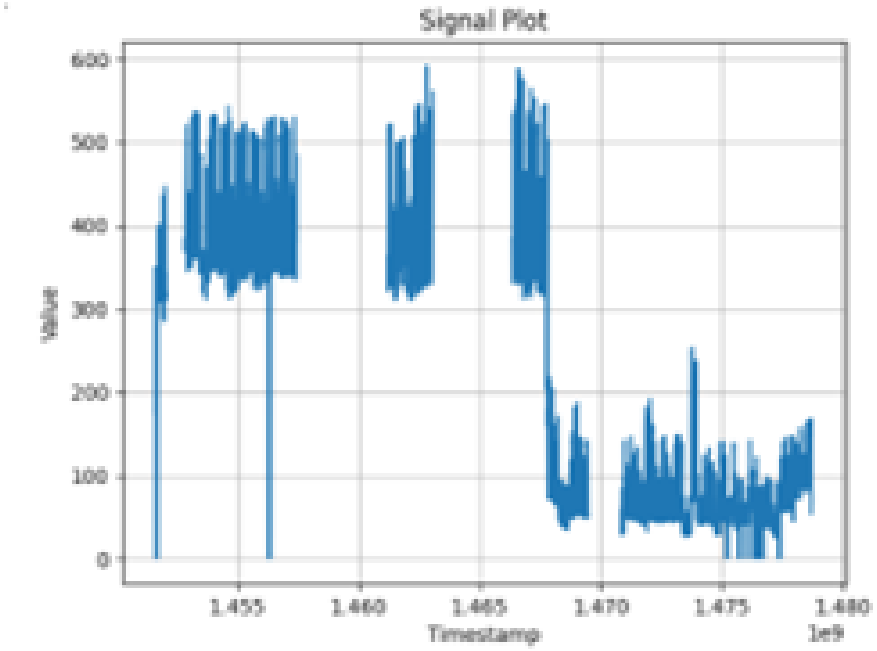


Figure 12: The time series plot of Building-107 values

The value vs time plot of these values showed several stretches of empty continuous patches with null values. As the absence of values was continuous over a large interval, imputation was not considered as a viable option. This is because time series models rely on temporal sequences and relations, and introducing a continuous stretch of imputed values would mean introducing a new, unwanted trend into the data, which can potentially harm the performance of recurrent neural networks constituting the time series network.

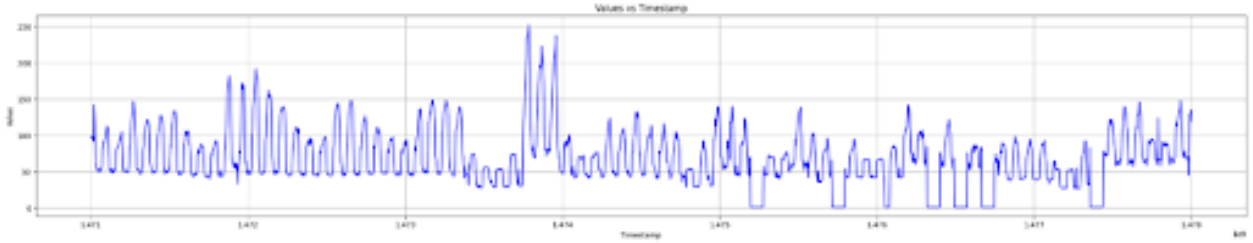


Figure 13: A continuous portion of building 107 data, on which the models were benchmarked

For this reason, we filtered the time series for these buildings to get the largest continuous stretch of non-null values, without stretches of NaN points in between. This prevents us from imputing large sequences of data that could introduce unwanted temporal relations while training the model.

The timestamp was converted from data-time format to the numerical Unix Epoch time format. The Unix epoch time converter converts timestamp to the number of seconds that have elapsed since January 1, 1970. As we are analyzing relations within the time series itself, the start date does not matter. The format helps in obtaining the timestamp in numerical form, and helps in ease of data handling and training of the time series models. The timestamps are all at equal intervals = 3600 seconds.

The indexes were then reset from those in the original dataframe containing all building points together.

The anomaly ground truth values were then aggregated into a ‘known_anomaly_build’ dataframe which had the start and end timestamps for continuous anomalous regions. This is to allow ease of computation of anomaly scores later on, where an overlap of segments between predicted and ground truth labels is used. A numpy array was also created with the series of ground truth labels.

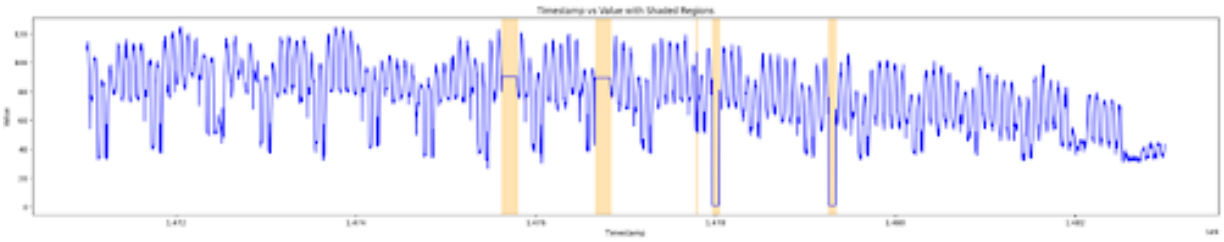


Figure 14: Anomalous regions shaded on a continuous portion of the Building-139 dataset, we used

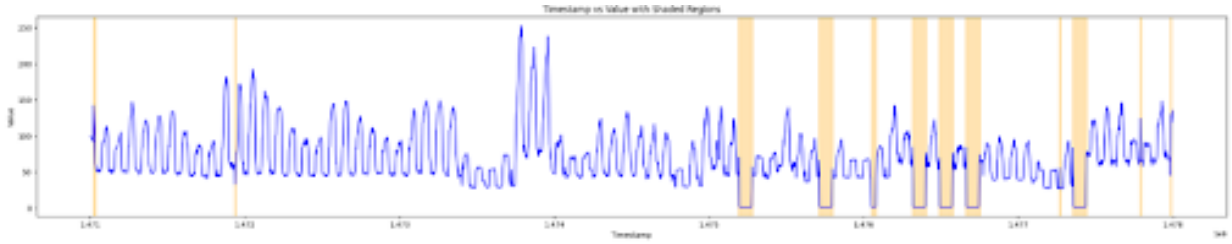


Figure 15: Anomalous regions shaded on a continuous portion of the Building-107 dataset, we used

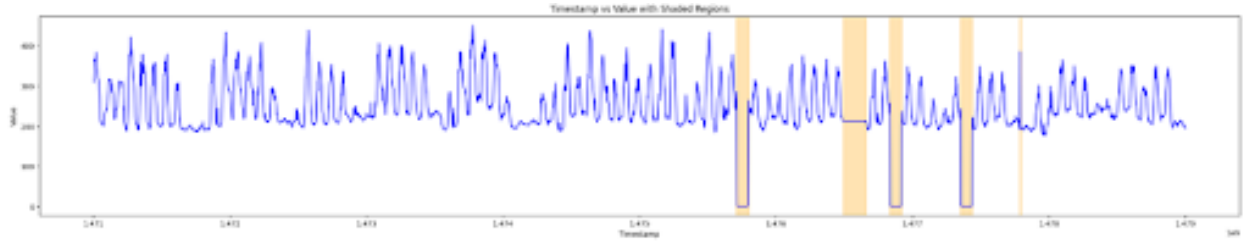


Figure 16: Anomalous regions shaded on a continuous portion of the Building-108 dataset, we used

Some EDA was carried out on the dataset to obtain better insights into the behavior of the data. The autocorrelation plot of building 139, shows high values at the initial lags, upto lag 5, indicating that the time series is highly dependent on recent past values. There is a decreasing trend followed by a slight increase around lag 15-20 which may suggest a periodic pattern.

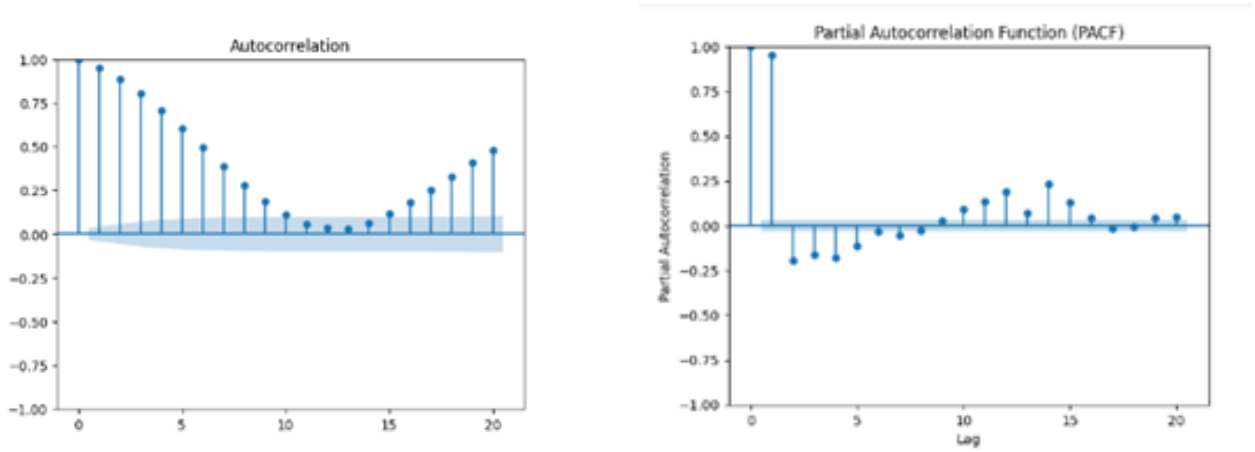


Figure 17: The ACF and PACF plots of the above portion of Building-139 dataset

This is further confirmed in the seasonal decomposition plot where we see repetition of behavior in small intervals. PACF shows strong partial autocorrelation at lags 1,2.

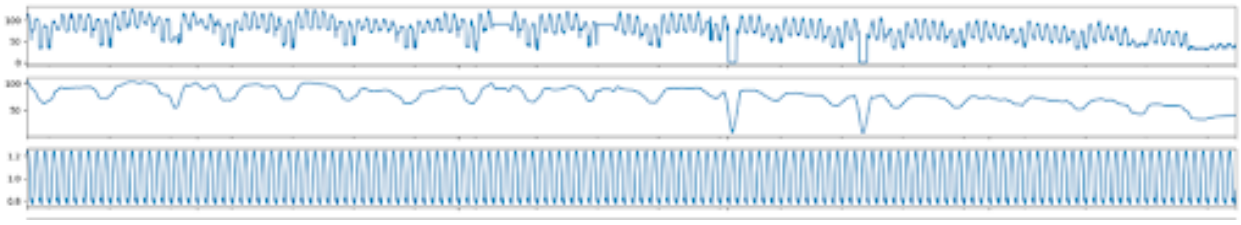


Figure 18: time series additive decomposition of the data

This reflects that the data is strongly dependent on its recent values in the past. It indicates a periodic seasonality at smaller intervals,which helps us select hyperparameters while training the model. For example, the frequent interval periodicity helped us decide for a "smaller window size", while training TADGAN since, as a short time period helped us capture all the necessary temporal features effectively, following which there were repetitions.

These insights were also useful while improving model performance.

4 Evaluation Metrics and Model Results

4.1 Choice of Error calculation Metrics

To calculate the error between the reconstructed and the original sequences, we used the error metrics used in the paper we used as reference.

- **Point-wise difference**: This is the most intuitive way to define the reconstruction error, which computes the difference between the true value and the reconstructed value at every time step:

$$s_t = |x_t - \hat{x}_t| \tag{6}$$

- **Area difference**: This is applied over windows of a certain length to measure the similarity between local regions. It is defined as the average difference between the areas beneath two curves of length l:

$$s_t = \frac{1}{2 \cdot l} \left| \int_{t-l}^{t+l} x^t - \hat{x}^t \, dx \right| \tag{7}$$

  Compared with the pointwise difference, the area difference is good at identifying the regions where small differences exist over a long period of time. trapezoidal rule is used to calculate the definite integral in the implementation.

  We mostly used DTW in finding the error between original and reconstructed sequences.

  DTW's ability to accurately calculate temporal alignments makes it a suitable tool for anomaly detection applications, as evidenced in several studies.

- **DTW(Dynamic Time Warping)**: We have two time series $X = (x_{t-1}, x_{t-l+1}, \ldots, x_{t+l})$ let it be the given time-series and $X' = (x_{t'-1}, x_{t'-l+1}, \ldots, x_{t'+l})$ be the reconstructed one. Let $W \in \mathbb{R}^{2l \times 2l}$ be a matrix such that the (i, j)-th element is a distance measure between x_i and x_j, denoted as w_k. We want to find the warp path $W^* = (W_1, W_2, \ldots, W_K)$ that defines the minimum distance between the two curves, subject to boundary conditions at the start and end, as well as constraints on continuity and monotonicity.

  The DTW distance between time series X and X is defined as follows:

$$s_t = W^* = \text{DTW}(X, \hat{X}) = \min_W \left[\frac{1}{K} \sqrt{\sum_{k=1}^{K} w_k} \right] \tag{8}$$

  Similar to area difference, DTW is able to identify the regions of small difference over a long period of time, but DTW can handle time shift issues as well.

Due to these superiorities of DTW above the aforementioned 2 metrics, it is used mainly throughout the experiments.

4.2 Choice of Evaluation Metrics

AUC-ROC score [Area Under the Curve of Receiver Operating Characteristic] is used to evaluate the models. It enables us to check the True Positive Rate and False Positive rate by the model at various thresholds, from very low to very high. AUC-ROC represents the area under the ROC curve, obtained by plotting false positive rate (along x axis) and true positive rate (along y axis).

The baseline score is 0.5, which essentially suggests random guessing to classify. So an auc-roc score of a model above 0.5 is preferred. Higher the score the better is the model.

Other than this, we have used the best optimal threshold from the plot, via Youden J statistic, as shown in Eq. (9).

$$\mathrm{J} = \text{sensitivity} + \text{specificity} - 1 \tag{9}$$

is the distance between the ROC curve and the "chance line" - the ROC curve of a classifier that guesses randomly. The optimal threshold is that which maximizes the J Statistic.

Using this threshold, we have found the anomalous regions in most of the datasets.

After this, we calculated the F1 score, accuracy, precision, recall in the method described in the research papers [Orion library]. These are calculated based on the overlap between predicted anomalous regions and the actual anomalous region and the time segment length of the predicted and actual anomalous regions.

As this method considers the anomalies not just some independent discrete points, rather treating them as time segments, this process of metric evaluation was chosen.

Average Precision Score is the average of precisions of the model on dataset, for different thresholds (from very low to very high). So it is also a threshold independent metric.

The intuition is the following: since PR AUC focuses mainly on the positive class (PPV and TPR) it cares less about the frequent negative class. If we care more about the positive class and hence PPV and TPR we can go with Precision-Recall curve and PR AUC (average precision). This metric is preferred when the number of classes are heavily imbalanced [7] .

AUC-PR is the area under the precision recall curve and is the same as average precision score.

4.2.1 Choosing AUC-ROC to find Threshold

As we did not want the model to give false negatives [miss the anomaly].Failing to detect anomalies may lead to disasters in industry machines, we mostly used the AUC-ROC curve to find the best threshold, which also tries to keep recall at the best value. False positives can create unnecessary alarm, but will ensure safety.

4.2.2 Using Average precision Score and AUC-ROC score to judge model's performance

- For all labeled energy datasets, we found the best average precision score of each model, to assess the precision and used AUC-ROC score to get the optimal threshold [balancing precision and recall (mainly)].
- For some datasets (eg: NASA p-1,p-15, wind turbine-22 SCADA data) we have shown the AUC-PR curve, area and also the anomalous sequences found, using the threshold from PR-curve, maximizing the precision.

4.2.3 Weighted Segment

Weighted segment-based evaluation is a strict approach that weighs each segment by its actual time duration. It is valuable when we need to detect the exact segment of an anomaly without any slack. The process begins by segmenting the signal into partitions based on the ground truth and detected sequences. Next, it performs a segment-to-segment comparison, recording TP/FP/FN/TN accordingly. The overall score is then weighted by the duration of each segment. This operation is visually summarized by the illustration below. An interesting edge case of this approach is when the signal is regularly sampled, it becomes equivalent to a sample-based evaluation approach.

4.2.4 Overlapping Segment

This is a more lenient approach of evaluation. It takes the perspective of rewarding the model if it manages to alarm the user of even a subset of an anomaly. The idea of this approach is that behind an anomaly detector there are domain experts who monitor the signals. If an alarm is raised then the user will investigate the alarm. If the model partially identifies the anomaly, then the user will be able to look at the entire anomaly because it drew attention to its location.

Figure 19: Time-series with the labels and predicted anomalous region by model [1]

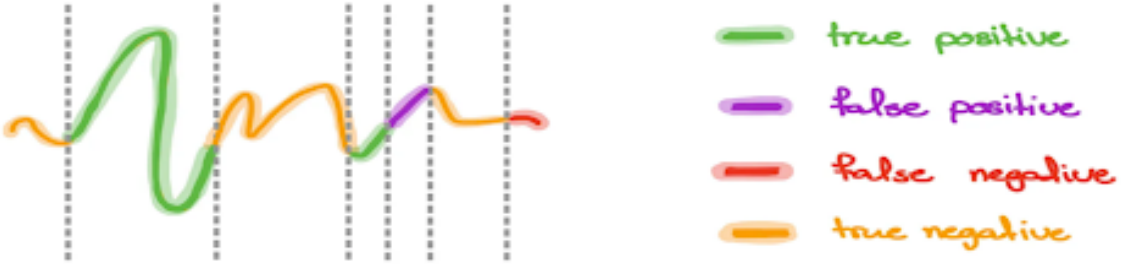


Figure 20: Weighted segment approach. Each vertical line depicts a partition. Each partitioned segment will be evaluated into a TP/FP/FN/TN based on the comparison of ground truth and detected segments. [1]

Hence, the anomaly is detected and the model is rewarded. This approach records (1) TP, if a ground truth segment overlaps with the detected segment. (2) FN, If the ground truth segment does not overlap any detected segments. (3) FP, If a detected segment does not overlap any labeled anomalous region. This can be summarized by the following figure.

Figure 21: Overlapping segment. Each detected anomaly will be assigned to either TP/FP, and missed ground truth anomalies will be assigned to FN. [1]

An example weighted segment approach:

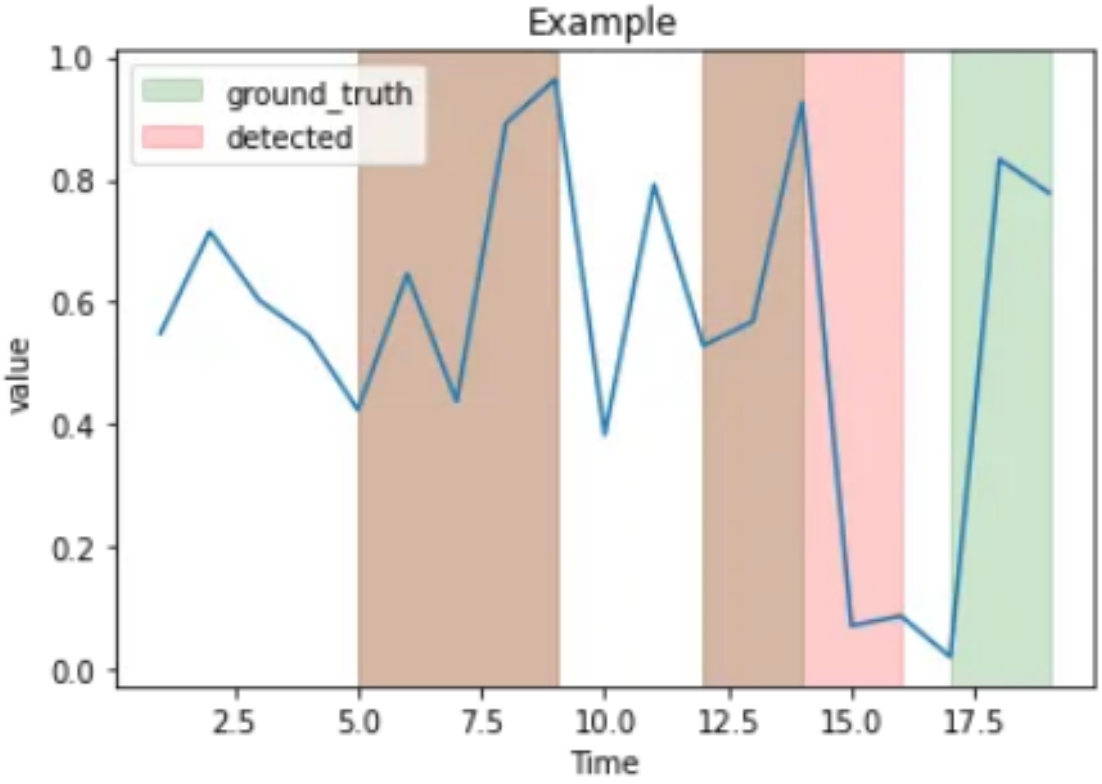
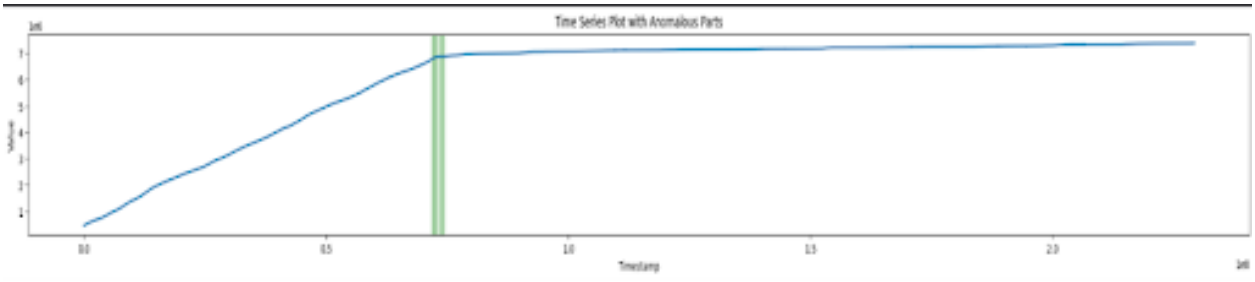


Figure 22: A sample time series data where weighted segment approach is used [1]

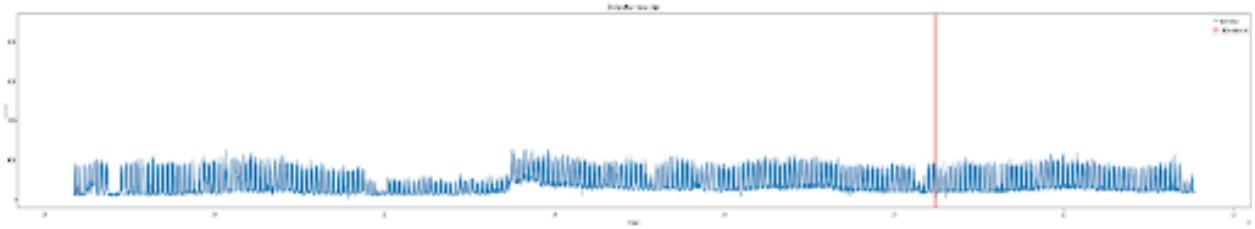
Using the weighted segment approach, we can see that we detected the majority of the anomalous interval, which if we are to look at the problem from a time point of view, we notice that we managed to detect (4 + 2) seconds of true positive, 2 seconds of false positive, and 2 second of false negative. Putting these together into our equation will yield precision = 0.75, recall = 0.75, and an F1 Score = 0.75.

4.3 AER Model predicted anomalous segments

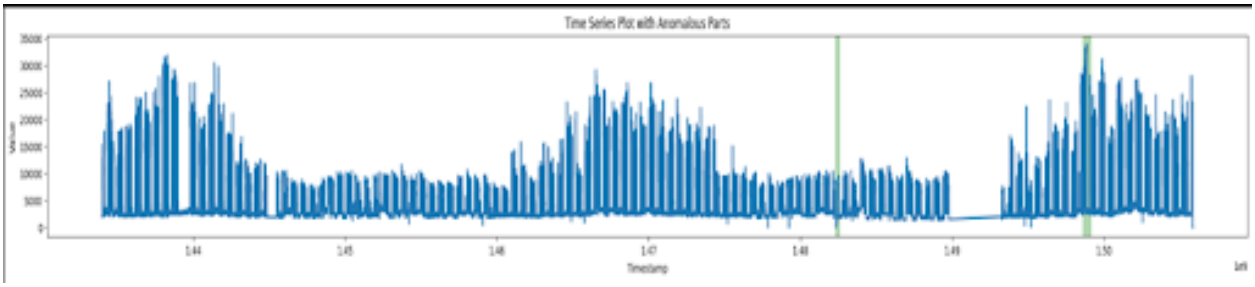
The Plots of Predicted Anomalous Regions on the Power Law Anomalies Dataset are shown in Fig 23. These plots are obtained after using Orion's find anomalies function.



(a) On meter id 36_9688 (captured the sudden change in trend correctly)

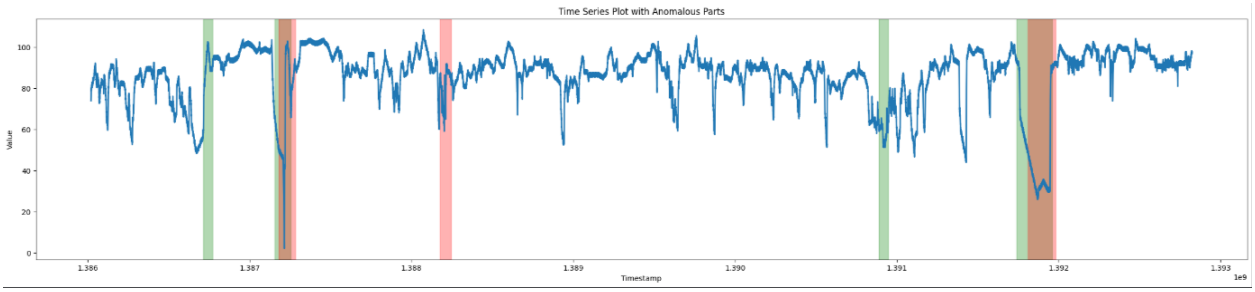


(b) On meter id 234_203 (just able to capture the globally high anomaly)

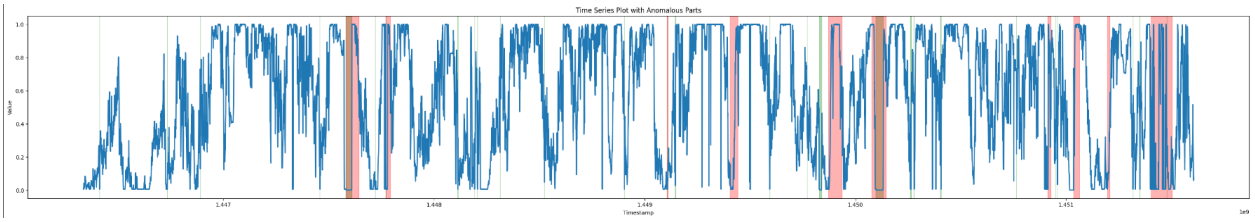


(c) On meter id 334_61 just able to identify the globally high suddenly increased part as anomalous

Figure 23: Plots showing different meter anomalies



(a) The anomalies, Labelled:green, predicted: red, using Orion's find anomalies function

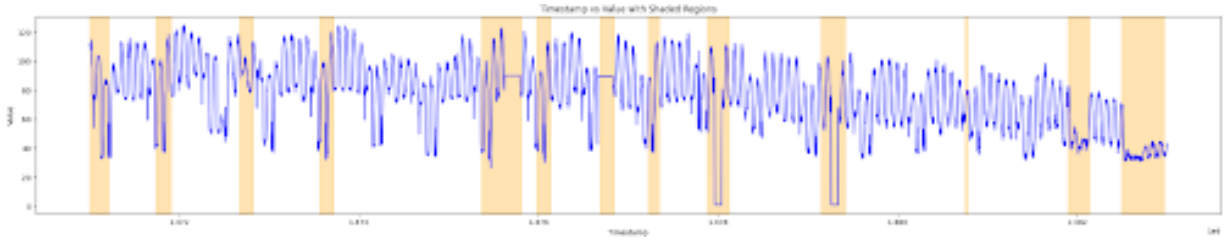


(b) The anomalies, Labelled:green, predicted: red, using the Youden J threshold

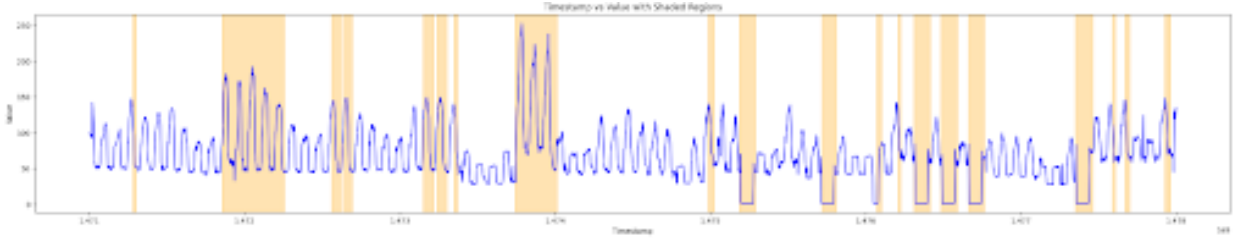
Figure 24: AER on (a) Machine Temperature System Failure and (b) SCADA turbine 22 Dataset

4.4 TadGAN Model predicted anomalous segments

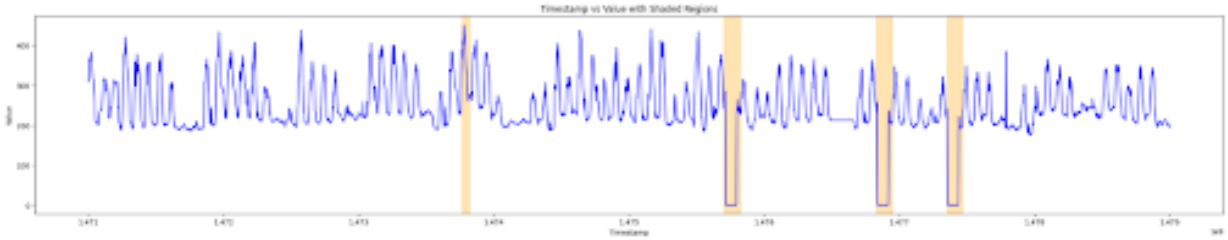
Fig. 25 shows the anomalous points of different buildings. Fig. 26 shows the changes when there is a sudden change trend shift.



(a) Signal with anomalous points plotted for building 138 (LEAD dataset)

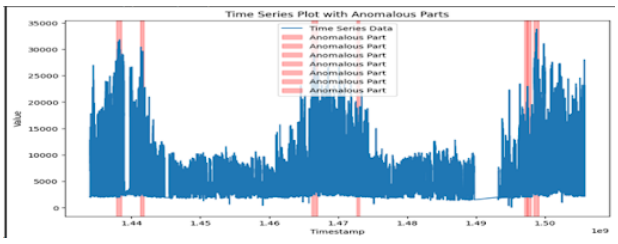


(b) Signal with anomalous points plotted for building 107 (LEAD dataset)

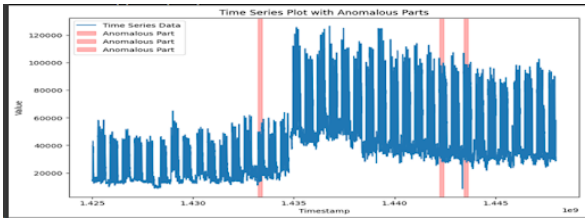


(c) Signal with anomalous points plotted for building 108 (LEAD dataset)

Figure 25: Signals with anomalous points plotted for different buildings (LEAD dataset)



(a) On meter id 334_61 after training TadGAN for 20 epochs



(b) On a part of meter_id: 234_203, where the trend shifts suddenly after training TadGAN for 20 epochs

Figure 26: Results after training TadGAN for 20 epochs on different meters

4.5 Comparative Results

For some of the data in Table 2, the performance is obtained after using double the time interval given in the dataset. For Table 3, all the results are calculated after running the model on each dataset 5-6 times with the best hyperparameters obtained by experimenting and using the best threshold from the auc-roc plot via Youden J statistic. Mostly, epochs of 5,10,15, or 20 were used. However, the sometimes model was set to run for 35 epochs, but due to early stopping, the training finished at different numbers of epochs with different run times but with comparable performances.

Table 2: Model performance on direct NASA telemetry signal, after the trivial preprocessing

Model	Dataset	AUC-ROC score	F-1 score	precision	recall	accuracy	Runtime (sec)
AER	NASA smap (s-1) (single var, trend shift anomaly)	0.936± 0.008	0.798± 0.007	1.000± 0.000	0.664± 0.011	0.926± 0.002	423.313± 125.29
	NASA msl (p-15) (single variable, point and trend shift anomaly)	1.000± 0.000	0.556± 0.017	0.405± 0.126	0.950± 0.000	0.976± 0.011	216.248± 2.488
	NASA smap (p-1) (single variable, contextual anomaly)	0.620± 0.029	0.213± 0.076	0.141± 0.064	0.505± 0.052	0.923± 0.037	747.79± 17.567
TadGAN	NASA smap (s-1) (single var, trend shift anomaly)	0.957 ±0.051	0.629 ±0.142	0.494 ±0.141	0.877 ±0.123	0.947 ±0.018	4771.452 ±68.792
	NASA msl (p-15) (single variable, point and trend shift anomaly)	0.997 ±0.006	0.565 ±0.125	0.41 ±0.132	0.95	0.979 ±0.01	2112.312 30.87
	NASA smapl (p-1) (single variable, contextual anomaly)	0.673 ±0.057	0.407 ±0.13	0.362 ±0.159	0.488 ±0.052	0.893 ±0.051	1035.57 ±24.193

5 Discussions

5.1 Limitations of TaDGAN

- **Training instabilities and High Computational Requirement :** The architecture of TadGAN consisting of two adversarial neural networks, being trained simultaneously with opposing objectives, with the generator trying to produce data indistinguishable from real instances and the discriminator trying to differentiate them, often leads to unstable training dynamics- in a zero-sum game, where the success of one network implies the failure of the other. This constant battle can cause instability- for example if the discriminator becomes very good at distinguishing real and fake samples at initial or intermediate training stages, it might provide gradients that are too extreme, leading to erratic updates for the generator. Moreover, most deep learning models are trained using mini batches where small groups are randomly sampled from the training set at every iteration. These random small groups, in some iterations, may not represent the overall true data distribution well, leading to significant variability of gradient estimates, and large oscillations in the adversarial training. This effect of noise can be tackled by averaging through larger batch sizes which lowers instability in training. However, larger batch sizes require more memory and computational expense per update, slowing down the

Table 3: AER and TadGAN performance on some energy datasets

Model	Dataset	Average precision	AUC-ROC score	F-1 score	Precision	Recall	Accuracy	Runtime (sec)
AER	LEAD (building 107)	0.353± 0.088	0.811± 0.163	0.405± 0.221	0.353± 0.194	0.519± 0.234	0.830± 0.143	77.32± 13.44
	LEAD (building 108)	0.525± 0.005	0.74± 0.018	0.511± 0.055	0.460± 0.099	0.586± 0.013	0.912± 0.017	158.271± 32.006
	LEAD (building 139)	0.269± 0.158	0.902± 0.027	0.309± 0.089	0.229± 0.101	0.520± 0.023	0.891± 0.033	117.1± 421.30
	Machine Temperature Failure dataset	0.608± 0.050	0.923± 0.006	0.326± 0.031	0.262± 0.187	0.676± 0.227	162.588± 9.675	567.387± 261.8
	WTPHM labelled SCADA windturbine data turbine -22	0.2625± 0.079	0.873± 0.006	0.240± 0.095	0.145± 0.011	0.701± 0.009	0.922± 0.006	391.306± 6.177
	WTPHM labelled SCADA windturbine data turbine -21	0.168± 0.010	0.350± 0.050	0.425± 0.010	0.274± 0.008	0.948± 0.015	0.409± 0.026	490.991± 31.288
TadGAN	WTPHM labelled SCADA windturbine data turbine -22	0.130 ±0.096	0.853 ±0.023	0.172 ±0.023	0.096 ±0.014	0.779 ±0.058	0.868 ±0.022	1097.813 ±5.468
	WTPHM labelled SCADA windturbine data turbine -21	0.276 ±0.0419	0.638 ±0.08	0.563 ±0.043	0.414 ±0.043	0.881 ±0.018	0.673 ±0.045	484.024 ±5.744
	Machine Temperature Failure Dataset	0.680 ±0.105	0.91 ±0.041	0.536 ±0.154	0.416 ±0.154	0.803 ±0.031	0.89 ±0.075	6271.218 ±143.715
	LEAD(building id 107)	0.327 ±0.029	0.88 ±0.011	0.426 ±0.041	0.286 ±0.034	0.890 ±0.037	0.776 ±0.040	156.196 ±2.876
	LEAD(building id 108)	0.617 ±0.005	0.83 ±0.008	0.642 ±0.021	0.707 ±0.044	0.588 ±0.007	0.934 ±0.011	194.278 ±18.041
	LEAD(building id 139)	0.317 ±0.075	0.847 ±0.045	0.247 ±0.044	0.143 ±0.029	0.9225 ±0.026	0.747 ±0.061	455.941 ±7.220

training at each iteration.

Moreover the two neural networks have to iteratively adapt to each other's feedback in every training step, following which the generator creates new instances with updated parameters and discriminator tries to improve on its sensitivity to avoid being fooled by the generator. This constant back and forth process of adversarial feedback, generation and discrimination, is computationally expensive, leading to slow training periods, especially in large datasets where many points have to be analyzed and reconstructed. For example , the SCADA dataset with about 35,000 points took 4000 seconds to train. Time series energy anomaly datasets are taken from electric smart grid meters which generate readings every hour, leading to a huge amount of time series data. The extensive time and computational power required by the TadGAN model means a longer deployment time.

This is a heavy drawback, especially from an industrial perspective. Industrial anomaly detectors require rapid detection of anomalies in dynamic energy data streams from different machine components in order to ensure

timely intervention. A model that takes a longer time to flag anomalies may be disadvantageous because it delays the identification of potential issues, resulting in prolonged operational disruption, safety hazards, machine damage and higher maintenance costs.

- **False positives :** The underlying architecture of the generator in TADGAN is based on an encoder decoder structure, where features are mapped into a low dimensional latent space and then reconstructed to recreate a fake instance. While feedback of the discriminator does help in better mapping of features into this latent space, there might still be some nuanced complex temporal correlations which cannot be easily compressed and recovered during the encoding and decoding phase, limiting the model's ability to capture some finer patterns in the data. This leads to many false positives in the outcomes on various datasets, due to the model's lack of ability in learning the underlying temporal behavior controlling those data points and a consequently high error when the real and reconstructed signals are compared. This raises false machine malfunction alarms, with unnecessary shutdowns or inspections that disrupt industrial processes and increase expenses.
- **Problems in detecting long contextual anomaly stretches:** The intuition behind reconstruction-based models lies in the fact that anomalies, lacking sufficient occurrence in regular data patterns, lose information when mapped to a lower dimensional space and thus, don't get reconstructed. This leads to high reconstruction errors when the original signal containing the anomalous region and the reconstructed one are compared, causing anomalies to get flagged.

  However, several anomalous regions involve large, continuous stretches of anomalous points, i.e. a continuous sequence stretching over several timestamps. Consequently, when passed into sliding windows for reconstruction, the continuous occurrence of these regions spanning several windows, forces the model to learn them as a normal temporal pattern, and thus reconstructs these anomalous regions as part of the normal signal. An example is the SCADA dataset, where a large stretch of flat anomalous region was reconstructed by TadGAN preventing it from being flagged as anomalous. While this problem can be partially handled using EMWA and a larger sliding window for detecting very long term trends, it can potentially have drawbacks. The use of EMWA and/ or window size results in a loss of sensitivity to short-term patterns in the data due to smoothing out(in EMWA) or dilution because of reduced influence of short patterns in a large window. This leads to an overtly smoothed out reconstruction, lacking information on the closer temporal correlations in the data. Due to this, a large number of regions with normal short term variations also lead to large errors, and are flagged as anomalous. We see this in the final plot of the SCADA dataset.

  This leads to a really high false positive rate, where the normal short term seasonal fluctuations frequently cause false alarms.

5.2 Limitations of AER

- The performance of AER (Auto-Encoder with Regression) is highly dependent on the quality and quantity of the available data. In real-world scenarios, obtaining high-quality data is often challenging. Limited data can lead to overfitting, where the model captures noise instead of meaningful patterns. Conversely, noisy data can degrade the model's ability to accurately detect anomalies. This limitation is particularly problematic in industrial settings where anomaly data is scarce or expensive to obtain. Consequently, the effectiveness of AER may be compromised in these environments, necessitating additional efforts in data collection and preprocessing.

  An illustrative example can be observed with the LEAD dataset, which is relatively small. Here, the model tends to overfit, yielding higher accuracy initially but potentially faltering as more data is introduced. Similarly, in the SCADA 21 dataset, a long segment of flatline—representing an anomaly—proved particularly problematic. The model's sensitivity to such anomalies led to a reconstruction that mirrored the flatline throughout (for reg_ratio values 0,0.5),until 'reg_ratio'= 1 was used, which resulted in a maximum AUC ROC of 0.32 and AUC PR of 0.16. Even doubling the time interval,which improved the score of AER,TadGAN on some other datasets, did not resolve this issue; instead, adjustments such as setting the regularization value to one, focusing solely on reconstruction-based anomalies, were necessary. This underscores AER's acute sensitivity to data quality issues.

  We have shown the performance of AER on this data in the hyperparameter sensitivity section with images.
- AER integrates prediction-based and reconstruction-based methods to leverage their respective strengths. However, this integration does not entirely eliminate the issues of false positives and false negatives. Prediction-based components often produce higher false positive rates, mistakenly identifying normal

variations as anomalies. This can lead to unnecessary alerts and potential operational inefficiencies. On the other hand, reconstruction-based components are more prone to false negatives, especially with point anomalies. This means genuine anomalies might be overlooked, posing risks in critical applications where undetected anomalies can lead to significant failures or hazards.

This can be observed in the P1 dataset where all the anomalies are contextual, when AER is applied to it the AUC ROC becomes 0.53 and the AUC PR becomes 0.21. This underscores that while AER attempts to mitigate issues inherent in both reconstruction and prediction-based anomaly detection, these challenges persist and are not entirely resolved.

- Time series data in industrial applications often exhibit non-stationary properties, where statistical characteristics such as mean and variance change over time. AER, despite incorporating techniques to manage some degree of variability, can struggle with highly dynamic non-stationary data. The model's performance can degrade in the presence of concept drift, seasonal variations, and other forms of non-stationarity, leading to missed or incorrect anomaly detections. This limitation necessitates the implementation of additional strategies, such as continuous model updating and adaptive learning techniques, to maintain performance in non-stationary environments. Without these strategies, the reliability of AER in detecting anomalies in changing data conditions is compromised.

5.3 Model performance comparison

From the experimental observations provided in Table: 1,2,3, we can declare the following

1. The AER model being auto-encoder, with simple architecture compared to TadGAN, is a lot faster than the latter.
2. AER shows overfitting to the given time-series data if trained for high number of epochs. So it is prone to over-fit. But no such overfitting is seen in case of TadGAN.
3. From Table: 1,2 it is evident that AER is better in detecting point and trend anomalies compared to TadGAN both in performance and speed.
4. While, from the same tables, we can see that TadGAN performs better on detecting contextual anomalies than AER, given enough epochs to train.
5. Both models are good in detecting point, 'sudden change in trend' anomalies.
6. Detecting contextual anomalies can be improved if we use the trend of time-series data to fit the models.
7. Eliminate noise in the data using EWMA, before fitting the data, to improve performance in detecting contextual anomalies (like- the long flat portion in SCADA wind turbine-21 dataset).
8. To obtain good performance on TadGAN, we need in general (15-20 epochs), whereas in AER, that requires (10 epochs on an average), used in predictive mode. As TadGAN only focuses on reconstruction, it needs a good number of epochs to reconstruct without overfitting. But AER, even when used in reconstruction mode, requires few epochs (5-10) and tends to overfit the series [evident from SCADA wind-turbine data]. For data of turbine-21, only the predictive mode of AER was able to detect anomaly in the long flat portion

(Note: predictive mode: "reg_ratio=1", reconstruction mode: "reg_ratio"=0)

5.4 Technique used to improve these model performance on time series data with contextual anomalies

We performed time-series decomposition and EWMA to improve performance. [6]

5.4.1 For P-1 dataset in NASA telemetry

The P-1 dataset was decomposed in additive manner and the trend contained point like anomalies at the regions where anomalies are present in actual data.

Time period=60 used for 1st decomposition.

To make the anomalous regions more prominent, the trend is again decomposed with time period=71

This trend was used to fit the model after the mentioned preprocessing techniques with the same labelled anomalous sequence as before. As a result, it produced better AUC-ROC scores, but degraded the AUC-PR [comparing with the performance of the model on same dataset also considering using double of the given time interval as a tuned

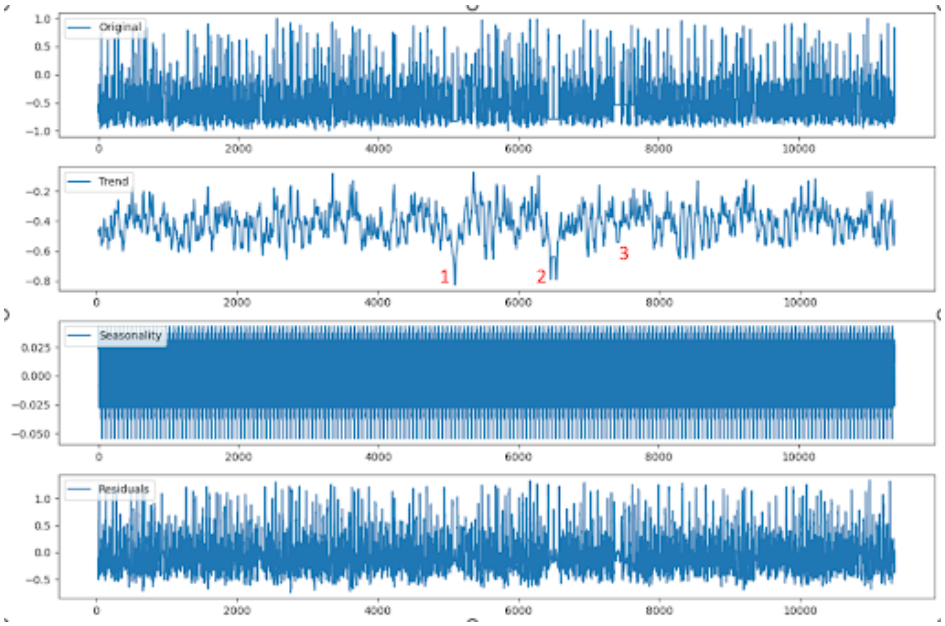


Figure 27: P-1 channel signal additive decomposition with numbered anomalous regions at trend

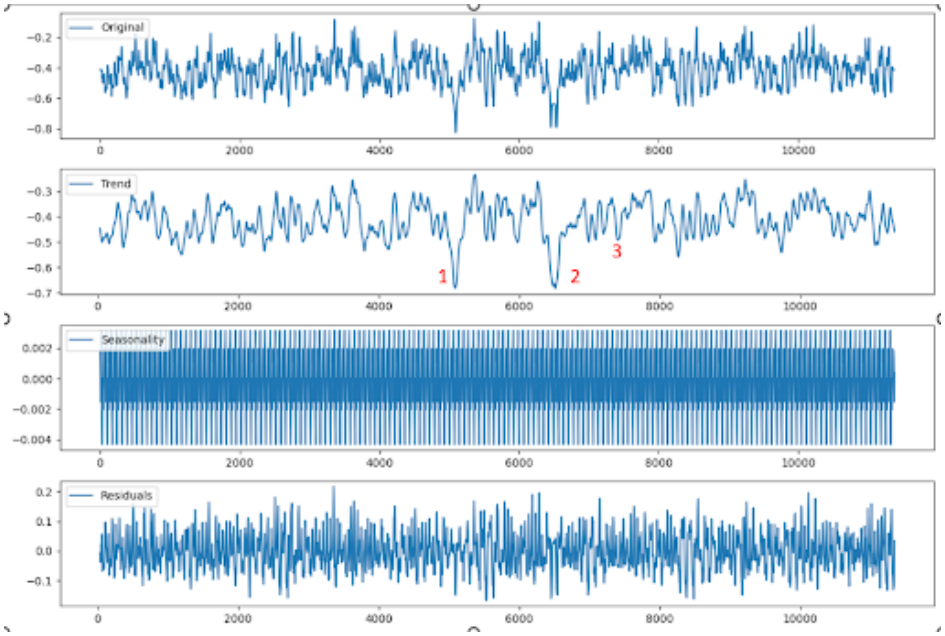


Figure 28: Additive decomposition of previous trend with numbered anomalous regions.

hyperparameter]. We took the hyperparameters with highest AUC-ROC and AUC-PR scores, using larger time-intervals, in table-4 and 5.

The results shown are obtained after hyper-parameter tuning and running the model multiple times on the datasets.

As we can see the average auc-roc, f1 score has increased for both models. But the variability has also increased slightly.

5.4.2 For SCADA-2015 Turbine-21 and Machine Temperature Failure Dataset

Used exponentially weighted moving average with smoothing parameter α=0.1, to eliminate unwanted noise. It drastically improved the performance of AER, based on AUC-ROC score and average precision score.

Table 4: Model performance on p-1 signal directly (with same labels)

Dataset	Model	AUC-PR	AUC-ROC	F-1 score	Precision	Recall	Accuracy	Runtime (sec)	Epochs (avg)
NASA smapl (p-1) (single variable, contextual anomaly)	AER	0.396 ±0.035	0.62 ±0.029	0.213 ±0.076	0.141 ±0.064	0.505 ±0.053	0.923 ±0.037	747.788 ±17.567	20
NASA smap (p-1) (single variable, contextual anomaly)	TadGAN	0.46 ±0.101	0.673 ±0.057	0.407 ±0.13	0.362 ±0.159	0.488 ±0.052	0.893 ±0.051	1035.57 ±24.19	10

Table 5: Model performance on p-1 signal using the 2nd order decomposed trend with same labels

Dataset	Model	AUC-PR	AUC-ROC	F-1 score	Precision	Recall	Accuracy	Runtime (sec)	Epochs (avg)
NASA smapl (p-1) (single variable, contextual anomaly)	AER	0.375 ±0.066	0.678 ±0.015	0.183 ±0.068	0.110 ±0.053	0.765 ±0.272	0.845 ±0.109	96.029 ±6.486	2
NASA smap (p-1) (single variable, contextual anomaly)	TadGAN	0.412 ±0.164	0.757 ±0.076	0.305 ±0.061	0.231 ±0.108	0.625 ±0.218	0.781 ±0.092	2211.216 ±12.547	20

The exponentially weighted moving average is calculated using the Eq. (10).

$$\text{EWMA}_t = \alpha \text{x}_t + (1 - \alpha)\text{EWMA}_{t-1} \tag{10}$$

Here,

- EWMA_t is the exponentially weighted moving average at time instant t.
- EWMA_{t-1} is the exponentially weighted moving average till time instant t-1
- α is the smoothing factor (0<α<1).

More the value of , more emphasis on present value of time series, more resemblance to original series and captures noise. Smaller value ensures the continuity of pattern of previous values, hence preserves smoothness.

Table 6: Effect of EWMA on AER & TadGAN on SCADA wind turbine-21 dataset

Model	EWMA	Best AUC-ROC	AUC-ROC score stats	Best average precision score	Average precision score stats	F1 score (orion)	Recall score (orion)	Precision score (orion)
AER	Without EWMA	0.40	0.350±0.050	0.179	0.168±0.010	0.436	0.96	0.282
	With EWMA	0.73	0.562±0.099	0.329	0.237±0.054	0.52	0.950	0.358
TadGAN	Without EWMA	0.71	0.638±0.078	0.313	0.276±0.042	0.596	0.904	0.445
	With EWMA	0.79	0.650±0.157	0.380	0.295±0.083	0.516	0.942	0.355

5.4.3 For SCADA-2015 Turbine-21 dataset

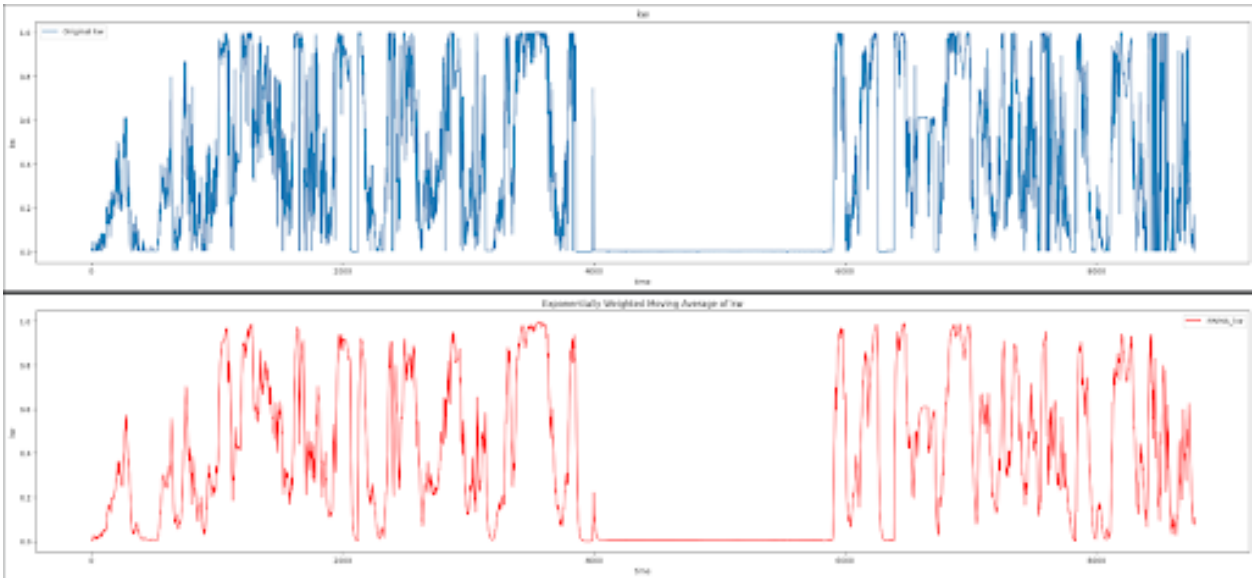


Figure 29: 1) The blue graph is original 'kw' column of the turbine 21 dataset [scada_21_df] 2)Red graph is the EWMA of blue graph (α=0.1)

- **For AER Model**

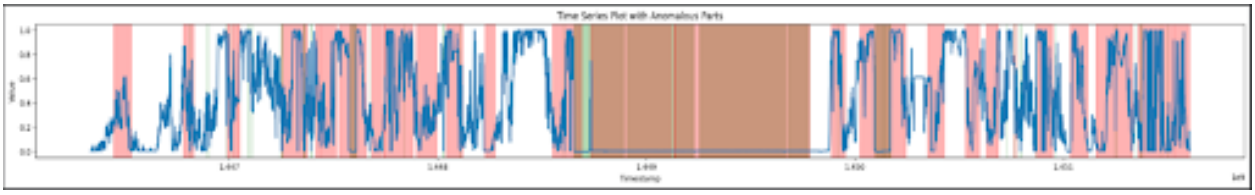


Figure 30: Anomalous regions predicted after using Youden J threshold on SCADA Turbine-21 data

In case of TadGAN, there was no improvement in the performance, when compared with the doubling of time interval. If we considered keeping same time interval as given in dataset, when not using EWMA, it's best AUC-ROC was 0.48 and 0.203 respectively. So compared to that state, the performance of tadgan has definitely improved.

- **For TadGAN Model**

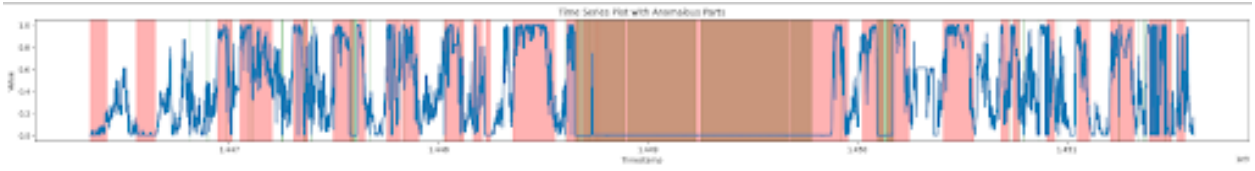


Figure 31: Anomalous regions predicted by TadGAN model after using Youden J threshold on SCADA Turbine-21 data

5.5 Improvement of AER performance in building 108 of LEAD dataset, with contextual anomaly continuous stretch [4]

In building 108, there was one contextual anomaly which involved a flat region(within the normal ranges of the data, but without regular oscillatory or seasonal behavior) stretching over a significant length.

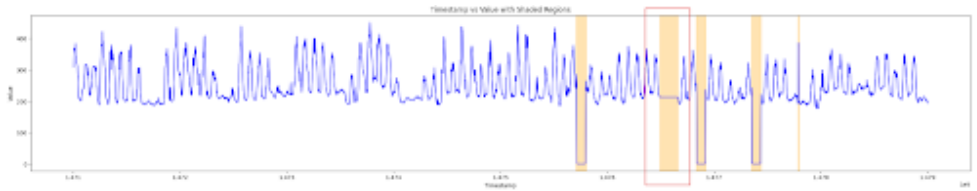


Figure 32: Original signal with marked anomalous parts, we are focussing on the rectangular-outlined part's reconstruction

In our initial approach of using AER on the univariate time series, the prediction and reconstructions were based solely on the current value of the time series. Due to this, the model still had a limited understanding of the expected or "normal" temporal dependencies in this region, due to the extended long stretch of the contextual error which confused it to inaccurately perceive it as part of the data's inherent behavior, leading to lesser errors, and non-detection of that anomalous region.

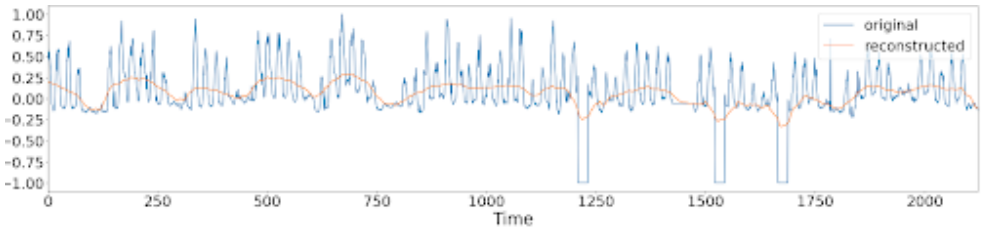


Figure 33: Poor reconstruction in univariate approach where it replicates abnormal flat behavior (in the previously marked rectangular region

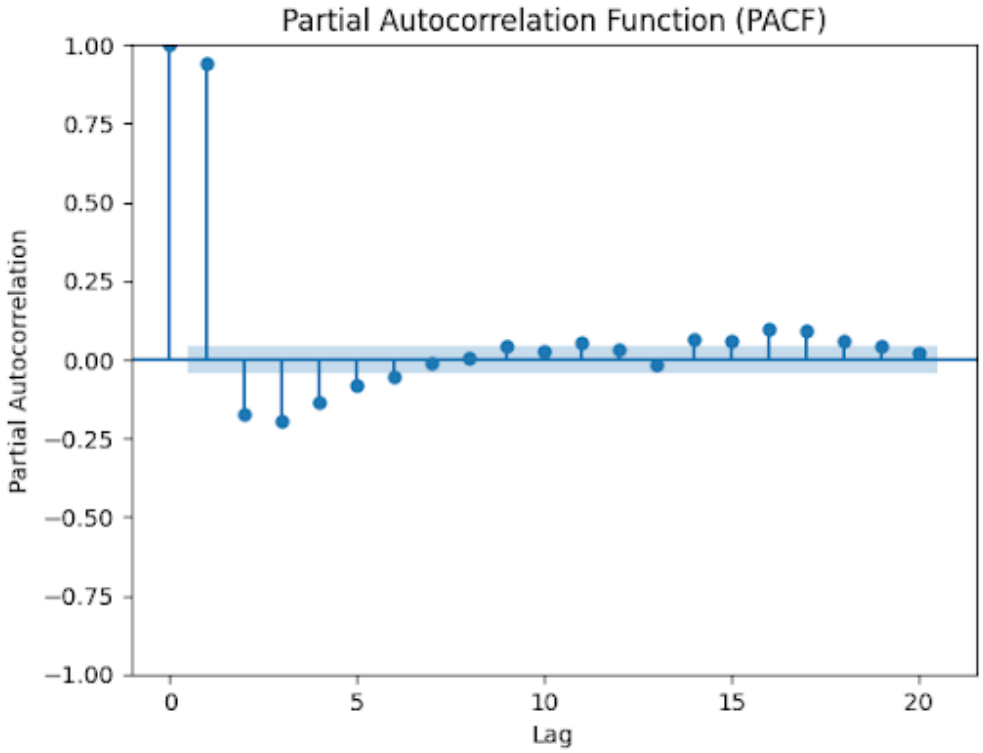


Figure 34: PACF of the function, showing strong correlation at lags 1,2

Partial Autocorrelation Function(PACF) helps in identifying the lag values which have a significant impact on the time series data. Our PACF shows strong correlations at lags 1,2 indicating that the past values present in the lagged version of the signal are important predictors of its current value.

We thus decided to include these lagged features(at lags 1,2) as part of the features, along with the current-value time signal, in the model input. By including these lagged time series as model inputs, we are essentially providing more contextual information about the time series dynamics, and emphasize on the temporal relations of its lagged versions (past trends). This multivariate approach helps the model make a more accurate prediction about the time series values, taking into account its lagged past versions, helping it gain more context.

	timestamp	value 0	value_lag1	value_lag2
0	1471010400	368.0	365.0	309.5
1	1471014000	362.5	368.0	365.0
2	1471017600	373.0	362.5	368.0
3	1471021200	383.5	373.0	362.5
4	1471024800	345.5	383.5	373.0
...	...	...	...	...
2215	1478984400	202.0	202.5	208.5
2216	1478988000	202.0	202.0	202.5
2217	1478991600	199.5	202.0	202.0
2218	1478995200	203.0	199.5	202.0
2219	1478998800	194.5	203.0	199.5

[2220 rows x 4 columns]

Figure 35: New dataframe with lagged time series included

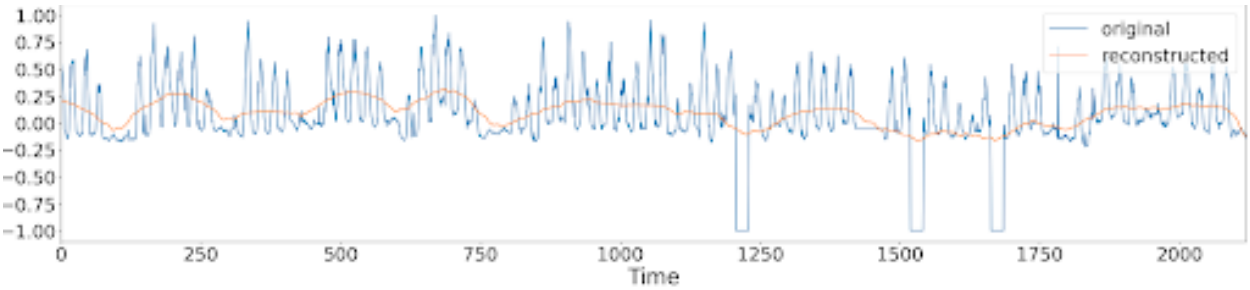


Figure 36: Better reconstruction in multivariate approach, the curve persists, even in the abnormal flat region, thus reflecting that a more reliable reconstruction preserving inherent temporal patterns was created

Moreover, we adjust the hyperparameter lambda to make the error function more prediction based, that makes it more sensitive to deviations in future values based on past behavior, thus more severely penalizing parts of the signal that behaved differently from their lagged version's behavior, helping to better detect contextual anomalies. Although this method led to detection of more false positives, it still helped us detect a contextual anomaly of significant length. From an industry perspective, an anomaly spanning for comparatively longer time periods could be very threatening , from an energy wastage or safety perspective. Hence this approach can be better for such purposes where our focus lies more on detecting as many anomalous regions as possible, and less on mitigating false positive alarms.

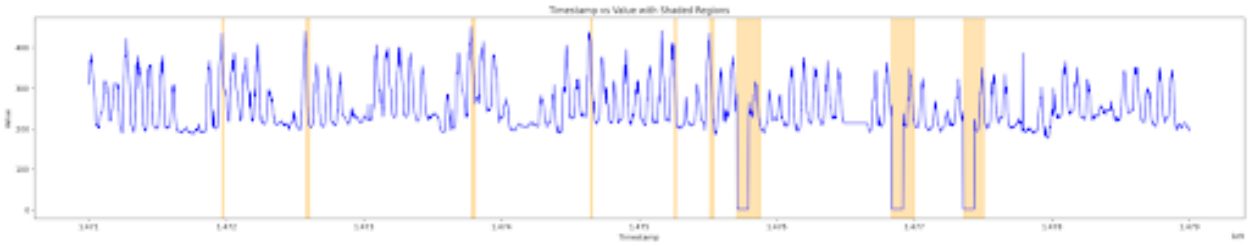


Figure 37: Anomalous regions detected in univariate approach

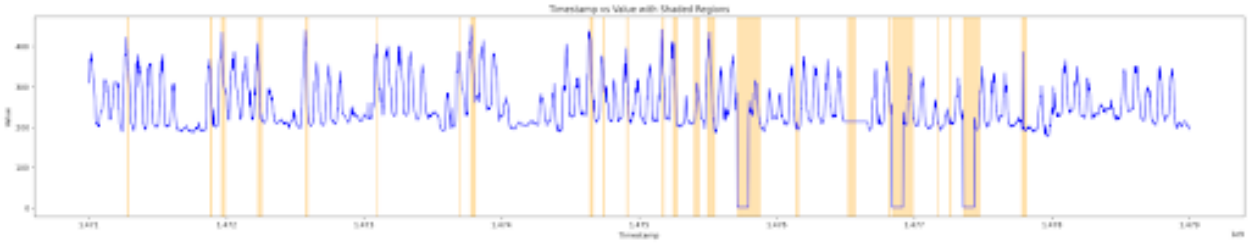


Figure 38: Anomalous regions detected in u multivariate approach. We notice how the lower plot has a shaded segment passing through the rectangular part, indicating that it detected the contextual anomaly as opposed to the previous one

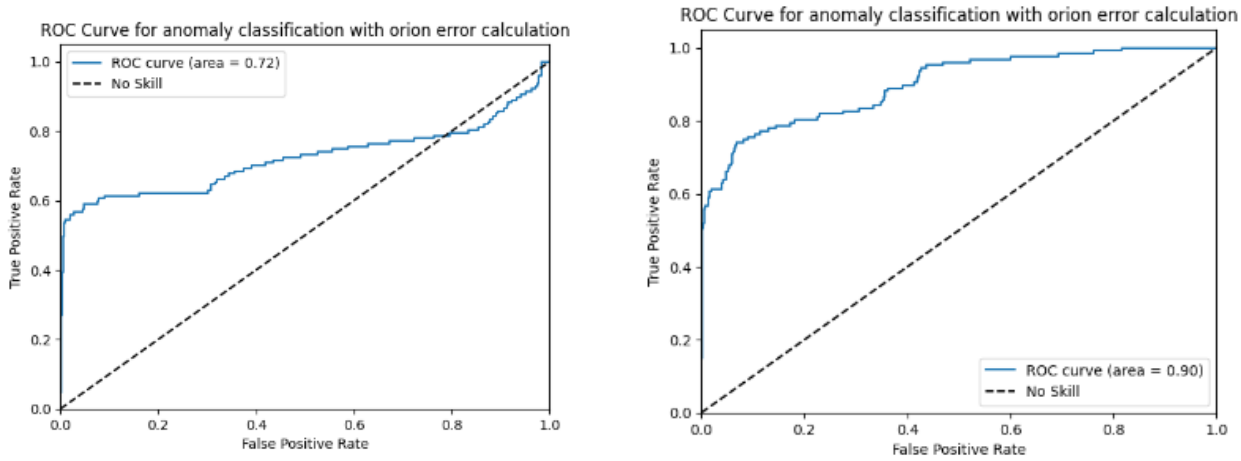


Figure 39: ROC-AUC of univariate (left) vs multivariate (right) approach

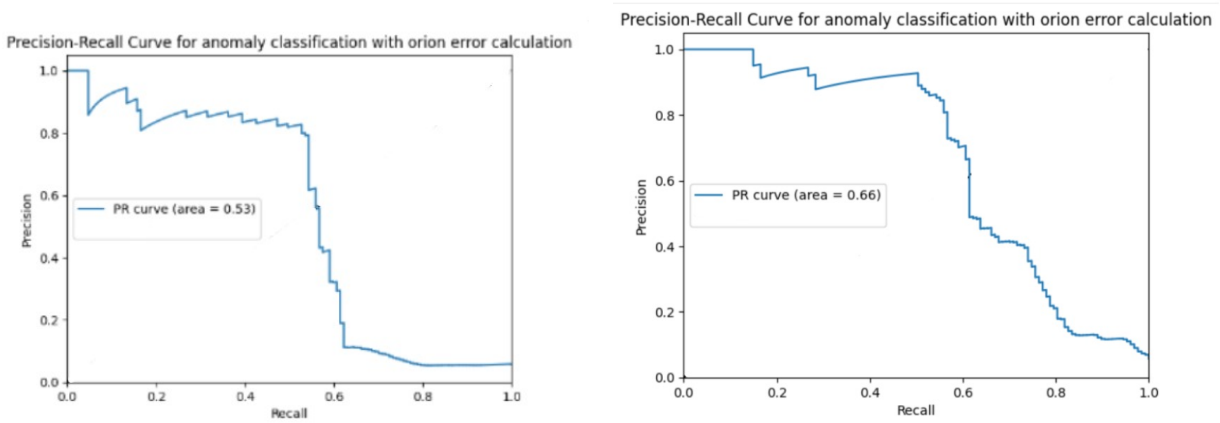


Figure 40: PR-curve univariate (left) vs multivariate (right) approach

Figure 41: Precision,Recall scores, obtained using the Youden J threshold. univariate (left) vs multivariate (right) approach

The increase in recall is due to improved detection of the contextual anomaly and thus a decrease in false negatives. Few extra points were also marked as anomalous in the new approach, which slightly decreased precision due to increased false positives. As previously explained, false negatives should be addressed more than false positives, thus the overall performance of the model, is still perceived as improved from an industrial perspective.

6 Further Discussions:

In the time-series anomalies, after reconstructing the time-series, the error is computed mainly in 2 ways [dtw and area]. Some observations regarding these error computation techniques are -

1. Though in general, the DTW computation of error between the 2 time-series is most efficient and produces good results, in some cases, the area based computation yielded better performance metrics [AUC-ROC and especially AUC-PR]. Also in most of the cases the area based error followed a very similar pattern/trend followed by the DTW error [seeing the graphs]. The cases where area based error gave better metric and also F1 score in orion calculation of confusion matrix, mostly had the model trained for comparatively low number of epochs [eg. TadGAN on p-1 data trained for 10 epochs]. Following are some examples.

   1) **With TadGAN on SCADA 2015 turbine-22 data:**

   Epochs:10 **Window_size:**100 input dataframe contained all normalized features and un-normalized counter columns

   Using DTW Error :

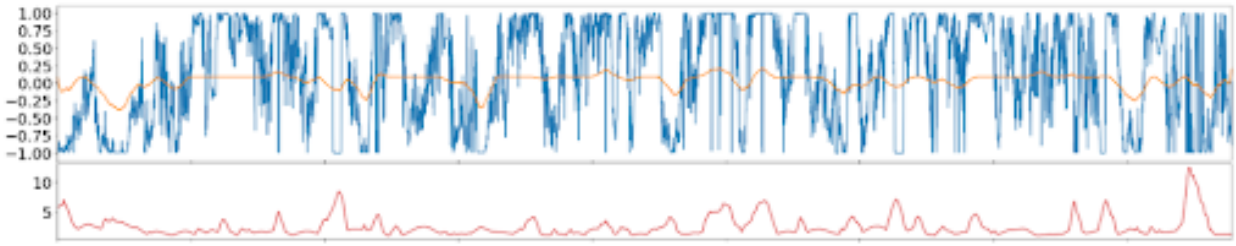


Figure 42: blue: original time-series orange: reconstructed series red: error

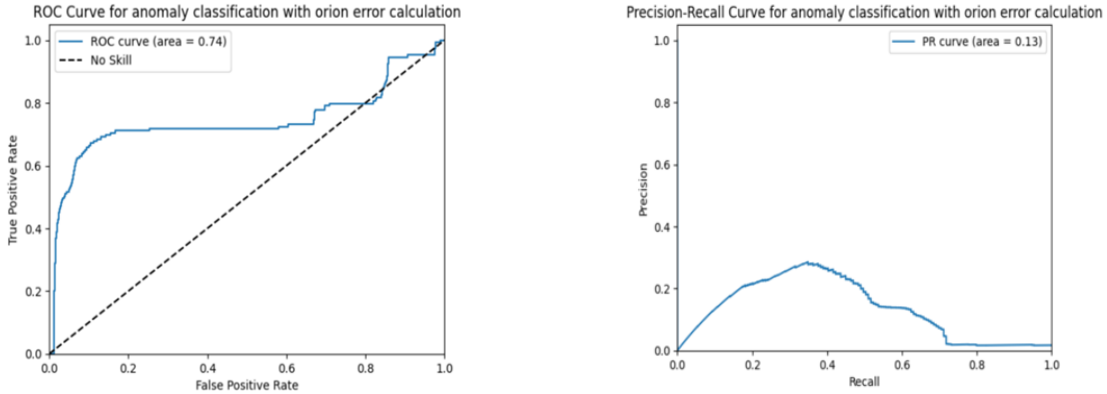


Figure 43: AUC-ROC and AUC-PR metrics for TadGAN using the above error

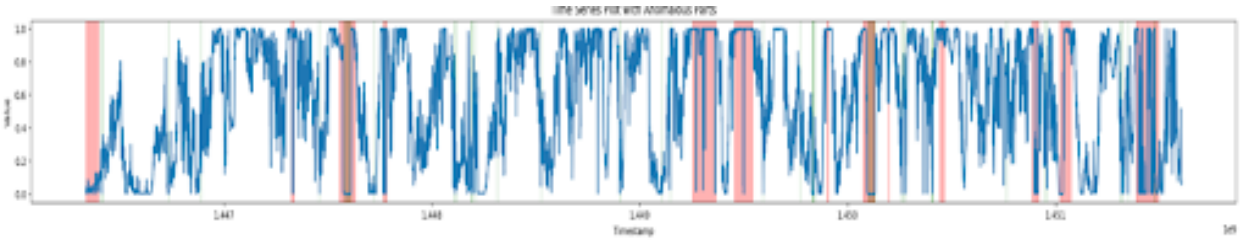


Figure 44: the predicted anomalous regions using Youden J threshold from ROC plot

```
Accuracy score = 0.885
F1 score = 0.167
precision:  0.09573670645198142
recall:  0.6684768950031141
```

Figure 45: The anomalous regions and score obtained after using optimized threshold from ROC plot of above

Using Area Error :

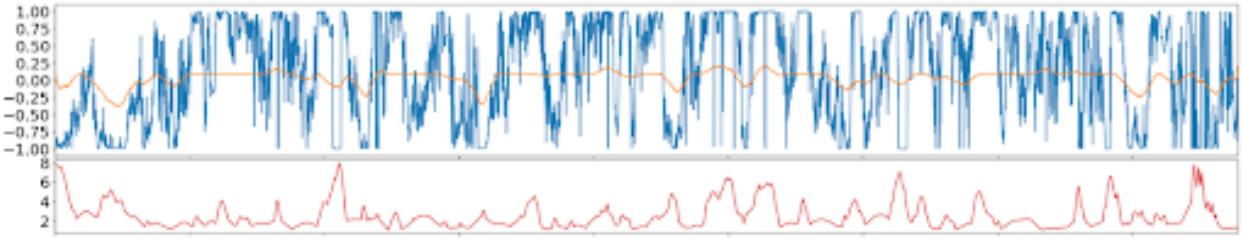


Figure 46: blue: original time-series orange: reconstructed series red: error

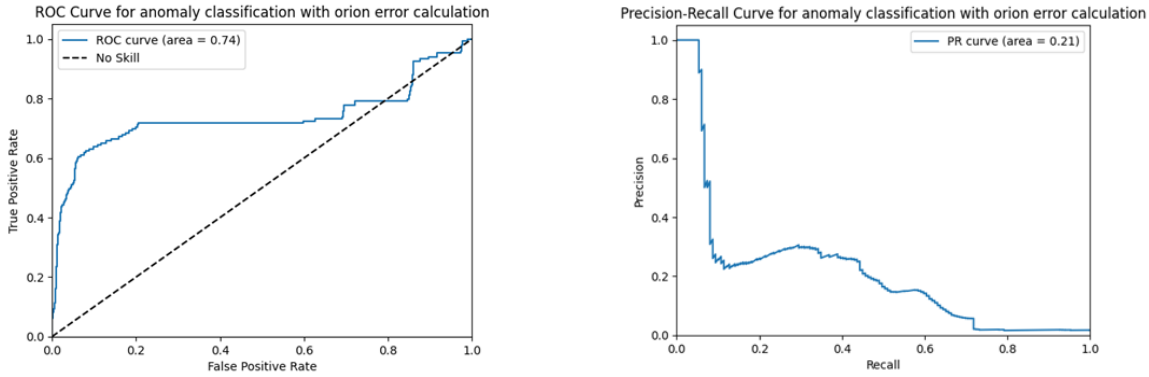


Figure 47: AUC-ROC and AUC-PR plots using Area based error

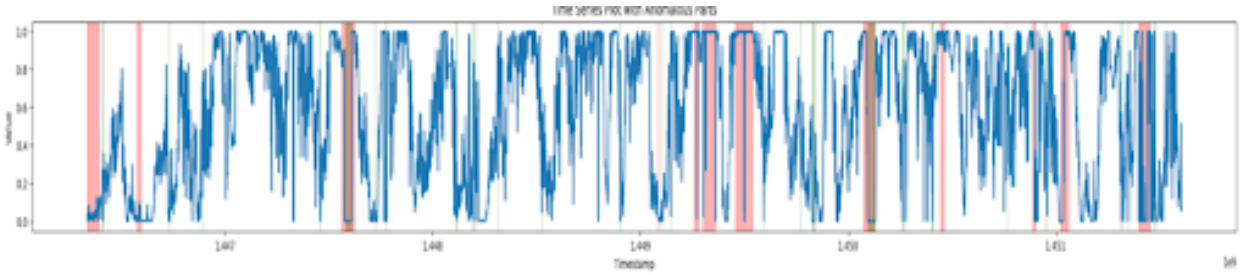


Figure 48: the anomalous regions and score obtained after using Youden J threshold from AUC-ROC plot of above

```
Accuracy score = 0.914
F1 score = 0.198
precision:  0.11829216645983148
recall:  0.6134707212867555
```

Figure 49: The classification score obtained after using optimized threshold from AUC-ROC plot of above

2) **With AER on P1 data:**
Using the parameters:—-
Epochs: 10 **reg_ratio:** 0.5 **window_size:** 250 **lamda:** 0.5 **DTW based error**

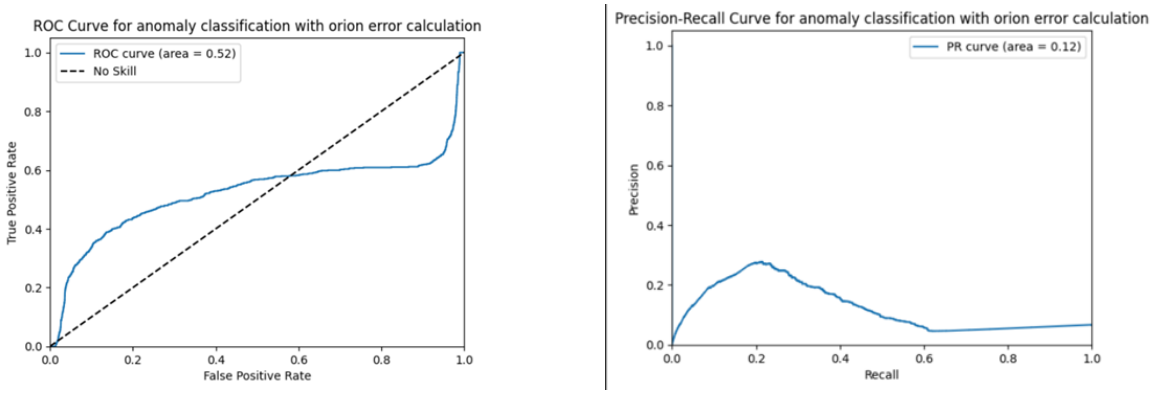


Figure 50: AUC-ROC and AUC-PR metrics for AER using the above error

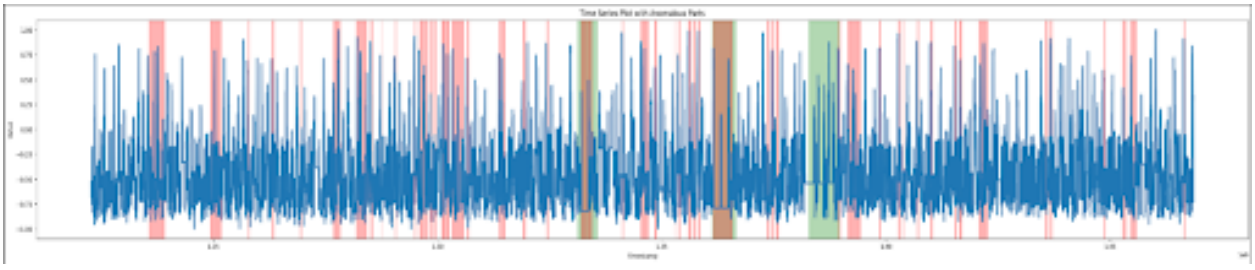


Figure 51: The anomalous regions predicted using Youden J threshold from ROC plot

```
Accuracy score = 0.840
F1 score = 0.102
precision:  0.05687751739016613
recall:  0.490000581018384
```

Figure 52: the classification scores obtained after using that optimized threshold from ROC plot of above

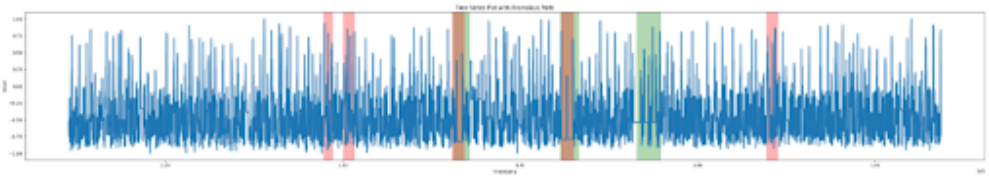
2. In time series, there can be many types of anomalies; here, we use Eq. (11) for the threshold determination.

$$\text{threshold} \approx \lambda \times \text{var}(X_{t-w:t}) \tag{11}$$

To find contextual anomalies. Orion library has the function to find anomalous regions using the errors, in a window-based method, using threshold similar to this. In this method, they create a window in the error series and whichever point has the error 4 standard deviations away from the mean of window is considered anomalous in that window. Though in our experiments, in most of the cases this did not work as good as the fixed threshold on error [threshold either from Youden J statistic from ROC curve or threshold with maximum precision from PR curve], in some cases it predicted anomalous regions better than both of the 2 ways.

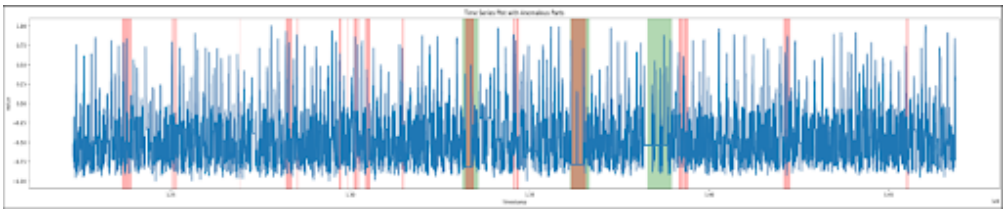
Example:–

AER model directly on P-1 dataset



```
Accuracy score = 0.892
F1 score = 0.408
precision:  0.41643453509151734
recall:  0.39973266988524897
```

Figure 53: Using orion's `find_anomalies()` function to predict the anomalous regions



```
Accuracy score = 0.924
F1 score = 0.200
precision:  0.12408746585371541
recall:  0.5100001134258997
```

Figure 54: Using optimized threshold from ROC-plot to predict the anomalous regions

A fixed threshold might miss anomalies in regions with naturally higher variability. Window-based thresholds consider the local context. Errors exceeding the standard deviation within a window are more likely to be anomalies compared to

points exceeding a global threshold. Also a fixed threshold is too sensitive to a flag point even with slightly higher error as anomalous, raising the number of false positives. The metric curves like ROC or PR curve are connected points obtained after using fixed thresholds. So in general, the window based thresholding method, accounting for standard deviation of error, is better to use in time-series anomaly detection.

7 Hyperparameter Sensitivity

In this section, based on the performance of TadGAN and AER on these datasets at different hyperparameter combinations, we comment upon the effect of some hyperparameters on the model performance and their sensitivities. We focus on the hyperparameters, window size, step size, interval, epochs, iteration critic, error type, batch size, error combination and regression ration in the following sections.

7.1 TadGAN

Table 7: TadGAN model performance on different hyperparameters

Hyperparameters	Dataset Used	Value of hyperparameter	AUC-ROC score	Average precision
Window Size	Turbine-21	80 100	0.12 0.47	0.2 0.13
	LEAD Building-139	48 200	0.8 0.45	0.12 0.05
Time-Interval	Turbine-22	600 1200	0.85 0.88	0.12 0.24
Epochs	Turbine-21	5 10	0.7 0.31	0.3 0.16
	Turbine-22	10 15	0.74 0.86	0.13 0.15
	LEAD Building-139	20 10	0.7 0.8	0.08 0.12
Iter-Critic	Turbine-21	5 10	0.63 0.65	0.33 0.34
Batch Size	LEAD Building-139	32 128	0.73 0.87	0.24 0.36
Step Size	Machine Temperature Failure	1 5	0.93 0.69	0.78 0.25

- **Window size**: A sensitive parameter. In some datasets, if kept small, the chances of flat reconstruction obtained after fitting the model on a time series is reduced, so the performance is improved [observed on turbine-22]. Also if reduced it may hamper the performance, [observed on turbine-21]. For the Lead dataset, a smaller window size gave better performance. This is because Exploratory Data Analysis of the datasets showed seasonality with a small time period, and thus the patterns of the signals repeated themselves at smaller intervals. Hence, in order to effectively reconstruct its pattern, a small window size of 20-50 was enough to capture the periodic temporal relations. A reconstruction on a window significantly larger than seasonality can degrade model performance. This is due to blurring of detailed patterns when analyzed over a larger context. Using larger windows can result in loss of information on short term trends and subtle inherent variations in the data,due an averaging effect, resulting in poor learning and reconstruction by the model. This results in poor understanding of trends, and poor results due to misclassification of anomalous and non-anomalous regions. Reconstruction of the building-139 time series for window size 200, the arrows show how averaging led to loss of information of the short term trend where the reconstructed signal values are expected to dip slightly, but are instead reconstructed at the same average height as the rest of the signal. This is because the rise and fall of these mean values of these oscillations was blurred(averaged out) when a larger window was taken, leading to loss of information.

  Reconstruction of the building-139 time series for window size 48,arrows now show a better reconstruction where information containing the expected trend of the dip in oscillating values is captured, since averaging in a very large window is prevented, thus leading to better learning and reconstruction.

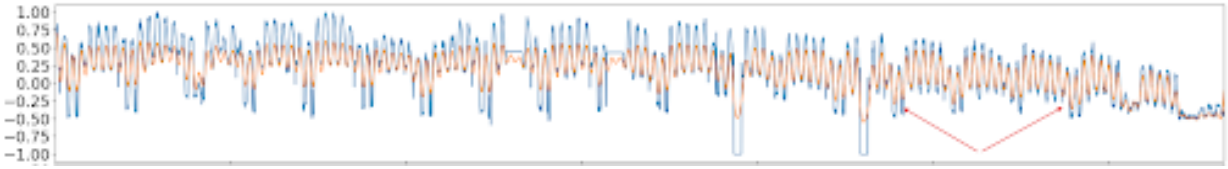


Figure 55: The original time series and reconstructions on LEAD Dataset building 139. Using window size: 48

- **Step size**: It refers to the number of time steps the sequence moves forward in order to form the next rolling window subsequence. We chose a step size of 1 for most of the datasets. This is because a smaller step size means more overlapping subsequences, and greater number of windows which provides a richer dataset for the model to learn from. As GANs require more training examples to stably learn the nuances of data distribution, and since our (some of our) datasets weren't very large, generating more training samples with smaller step size helped in effective model training without becoming too computationally expensive. Increasing the step size deteriorated the performance of model considerably, as evident from table data.
- **Interval**: This parameter value is selected based on what interval we want to sample the time series. If the time-series is composed of samples taken in a particular interval in timestamps, the interval is selected either equals to that or a value more than that [mean of the values at timestamps within that interval is used]. Even if the performance scores improved, the reconstructed series is somewhat more regularized when interval is increased, as the compressed series is also less noisy than the original series. So, chances of under-fitting is also more.This is evident from SCADA turbine-21 data. But still increasing it improves results, due to it's support in generalizing the data.

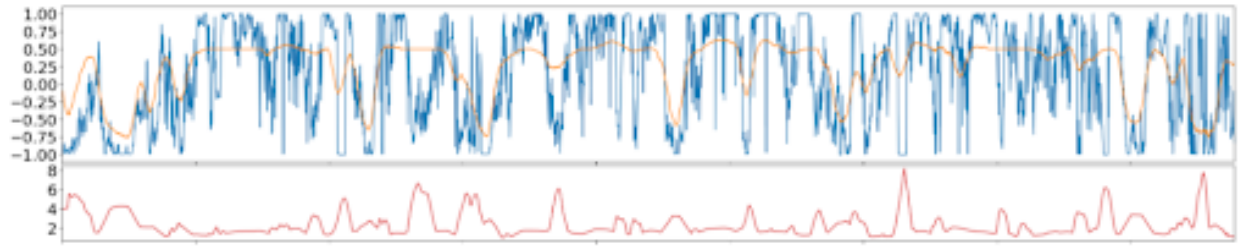


Figure 56: the reconstruction and error plot obtained after fitting model

- **Epochs**: As this is a GAN, and it has an inbuilt regularizing mechanism, so overfitting generally does not occur for this model. So in general from the reconstructions on time series plots, as the number of epochs is increased, the reconstructed series reaches close to actual time-series. But the same is not always true for change in performance metrics AUC-ROC and AUC-PR scores. If the reconstruction is very poor, most of the places in actual time series will be deviated from reconstructed series. So differentiating anomalous regions from normal regions will become difficult.If reconstruction very closely resembles the actual series, anomalous regions also produce less error. For some datasets, like SCADA turbine-21 data, even less number of epochs work better. In this dataset, training the model for 5 epochs gave better results than 10 epochs, consistently, keeping all other hyperparameters same. For some, like SCADA Turbine-22 data, it needs high epochs. In this dataset, training the model for 10 epochs gave better results than 15 epochs, consistently, keeping all other hyperparameters same. In general the required number of epochs to fit TadGAN on the datasets was mostly on the higher side,as it is a GAN based architecture. This is not a very sensitive parameter, rather it depends more upon other parameters. The same results apply for the LEAD dataset. Once hyperparameters were tuned for smoother and more effective learning for the generative adversarial neural networks, they converged to a stable solution within 10 epochs, and increasing beyond it, did not create much of a difference, in terms of reconstruction, since the necessary patterns were already learnt by the model. Hence we used a lesser number of epochs for lesser training time of the model.
- **Iteration Critic**: It refers to the number of iterations for the critic to train per generator update. By increasing it, the discriminator can become more accurate in distinguishing the fake and real time signals, which makes it learn the pattern more accurately before every generator update. This improved accuracy makes the critic provide more reliable gradients to the generator, due to more in-depth learning of the data's inherent pattern, leading to better updates in generator's behavior. In this way, the critic learns to provide better feedback, and thus the generator learns how to create realistic data more effectively. With more robust generator learning,

due to a more sensitive discriminator feedback per update, the generator creates more accurate reconstructions of the normal data, making it easier to detect anomalies with reconstruction errors and hence, improving performance, for the same number of epochs. This improvement in generator learning also helps it to converge faster to an accurate reconstruction of the signal for the same number of epochs, and can be attributed as the reason behind faster fitting time to the model. The optimal value was found to be 10, beyond which much improvement was not noticed, presumably because the critics were made sensitive enough for a robust learning of the overall time series temporal relations by the generator.

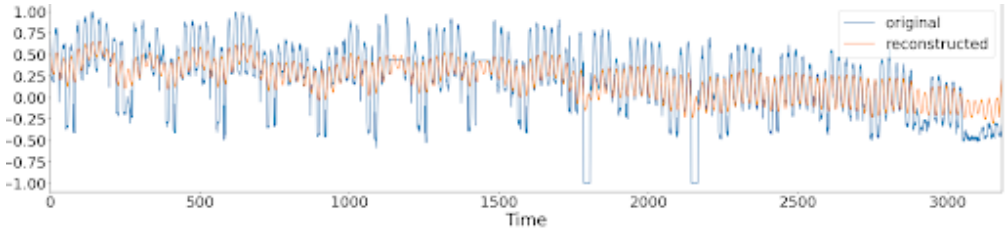


Figure 57: LEAD Building 139 with lower value(=5) of critic iterations

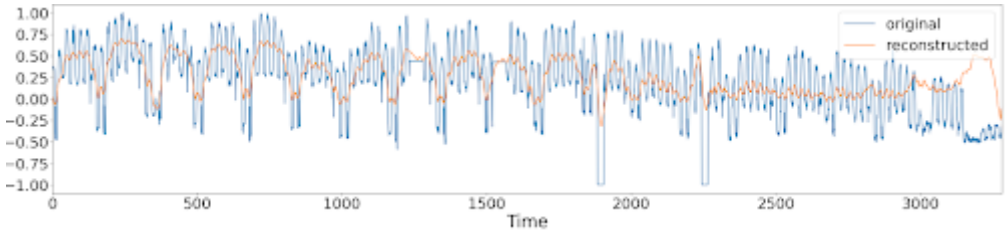


Figure 58: LEAD DATASET: BUILDING 139 with higher value(=10)

Plots of reconstructed signal without(upper) and with(lower) increased value of "critic iterations". It is noticed how the upper plot has simply captured an oscillatory behavior of the data, and replicated that, while the lower plot captures better temporal dependencies, highlighting trends and seasonal peaks better.

- **Error Type**: In most of the datasets, the DTW error gave slightly better results in terms of AUC-ROC and AUC-PR, compared to area. But in some cases using the area based error gave a much better AUC-PR, as discussed before. In very few cases, when the dataset contains only point anomalies, the point based error gave slightly better performance than area, but DTW gave best performance.
- **Batch Size**: GANs involve optimizing two adversarial networks simultaneously, and can be characterized by a min-max process, where the generator tries to minimize differences between fake and real instances while the discriminator tries to maximize it in its critic iterations. This can lead to oscillations in the training. This dynamic balance between the generator and discriminator can be disrupted in the presence of noisy updates, leading to large oscillations that destabilize and potentially harm model performance. While training with small batch sizes, noise can arise because small batches, randomly sampled at every iteration, may not represent the overall true data distribution well, leading to significant variability in gradient estimates, and oscillations in the adversarial training. Larger batch sizes help in averaging the noise in the gradients computed during backpropagation, which helps in more stable and smooth updates for the discriminator and generator. This helps more stable training, and lesser standard deviation in model outputs, and better convergence of the model. We also noticed a huge standard deviation for the former as training was unstable, due to the effect of noise in a small batch size.
- **Error combination** : The multiplication combination of errors (critic score and reconstruction error) gives better results compared to the convex combination of them,but the scores produced did not differ much.

7.2 AER Model

- **Window Size**: The window size determines the amount of historical data the model analyzes at once, impacting both its ability to capture patterns and its computational efficiency. A window size too small might miss long-term trends or cyclical patterns crucial for identifying anomalies. Such a window wouldn't capture the entire seasonal cycle, potentially missing anomalies related to extreme deviations from the usual seasonal

pattern. Conversely, a window size too large might include irrelevant data, making the model less sensitive to short-term anomalies. Hence, it might capture seasonal trends but miss sudden spikes due to equipment malfunctions. When we experiment with this on the LEAD dataset, we find the most noticeable impact it has is on the run time. A bigger window size takes more time, as expected. In our dataset, a bigger window size slightly performs better, which complements our EDA findings in autocorrelation.

- **Interval**: Tuning this hyperparameter, especially increasing it gave good results on most of the datasets. Ordinarily, in a time series this value is kept the same as the time interval between 2 successive instances in a time series. But if we make it twice, the average of 2 successive intervals is treated as 1 instance. On datasets like- NASA p1, Machine Temperature Failure datasets, using 2x of the original time interval improved the results. But on the SCADA turbine-21 dataset, the result remained almost the same, as depicted in Table 8.

Table 8: AER model performance on different hyperparameters

Hyperparameters	Dataset Used	Value of hyperparameter	Average precision	AUC-ROC score
Window Size	LEAD, building 139	50 200	0.260 0.363	0.766 0.915
Time-interval	Machine Temperature Failure	300 600	0.511 0.643	0.890 0.932
	NASA P1-dataset	21600 43200	0.210 0.359	0.513 0.661
	SCADA Turbine-21 dataset	600 1200	0.160 0.160	0.317 0.301
Step Size	LEAD, building 107	5 10	0.127 0.086	0.625 0.435
Batch Size	LEAD, building 107	32 128	0.264 0.205	0.822 0.786
Epochs	Decomposed P1 dataset	2 5	0.361 0.260	0.660 0.680
Regression Ratio	LEAD, building 139	0 1	0.416 0.148	0.905 0.535
	SCADA Turbine-22 dataset	0 0.5	0.589 0.231	0.860 0.873
	Machine Temperature Failure	0 0.5 1	0.401 0.641 0.170	0.762 0.930 0.570
Error Type	LEAD, building 107	Point Area DTW	0.238 0.546 0.273	0.824 0.910 0.838
Lambda	LEAD, building 107	0 1	0.491 0.251	0.899 0.831
Error Combination	LEAD, building 107	Sum Mult Rec	0.211 0.273 0.493	0.792 0.838 0.889

- **Step Size**: Step size, alongside window size, plays a crucial role in how AER (Auto-Encoder with Regression) processes time series data for anomaly detection. It determines how much the window "slides" forward when analyzing the data, impacting both computational efficiency and the model's ability to capture temporal dynamics. This can be a useful parameter when the data deviates from the general trend for quite some time. In our datasets, we observe this to be a sensitive parameter. This decreases the model performance by increasing it, as observed in Table 8. This shows that, in our dataset (LEAD), including more data from the previous window is better. This captures more context and can lead to smoother anomaly scores, potentially improving the detection of gradual changes. This again goes with our EDA findings.
- **Batch size**: Batch size is a crucial hyperparameter in training any machine learning model, including AER, for time series anomaly detection. It determines the number of data sequences processed by the model during a single training iteration. This impacts both the speed of training and the accuracy of the final model. A

smaller batch size requires more training iterations to process the entire dataset but can sometimes lead to better model convergence, especially for complex data or models. They also tend to be less susceptible to noisy data points within a single batch. A bigger batch size processes data faster by updating the model parameters with information from a larger chunk of data at once. However, they can lead to some issues. The model might get stuck in suboptimal solutions (local minima) during optimization. Also, the model might learn specific patterns from the batch that don't generalize well to unseen data. In our experiments on LEAD with varying batch size we notice the model performance increases with decreasing batch size, though it comes at the cost of computation time. This was as expected from the general trends.

- **Reg_ratio**: This sensitive parameter decides relative importance of reconstruction error and predictive error (forward and backward) during fitting the model. Using a value very close to 1 increases the predictive nature of the model, thus, focusing more on the previous trend in reconstructed time-series and after a low number of epochs (2-5), it's performance saturates. Better for datasets where anomalous regions are deviated from the normal trend or are made of sudden noise. Using value 0.5, equally gives importance to prediction and reconstruction equally. With this default value the best performance of the model is generally obtained at epochs (5-10). This default value gives good performance on an average. Using value 0, focuses on reconstruction completely. So it is advantageous to find out the contextual anomalies. Generally requires higher epochs (10 or more). This hyperparameter controls the relative importance of the prediction error compared to the reconstruction error in the final anomaly score. The AER model combines an auto-encoder, which learns the underlying structure of the data, and an LSTM regressor, which predicts future values. The regression ratio determines how much weight each error type contributes to the final anomaly score. High Regression Ratio emphasizes the prediction error. The model focuses on identifying deviations from the expected future values as potential anomalies. This might be suitable for data with complex temporal dynamics where anomalies manifest as unexpected changes in future behavior. Low Regression Ratio focuses more on the reconstruction error. The model prioritizes anomalies that make it difficult for the auto-encoder to accurately reconstruct the data, suggesting deviations from the underlying structure. This might be appropriate for data with a strong underlying pattern where anomalies primarily affect reconstruction accuracy. In our experiments we find this to be a sensitive parameter. It requires a perfect balance between 0 and 1 to be optimal. This complements the fact that our dataset of LEAD has both point and contextual anomalies, so ignoring either gives bad outputs. For the building 139 we observe the performance to greatly decrease for reg_ratio as 1, suggesting the contextual errors cannot get well represented, thus it needs more focus on the reconstruction error.
- **Error type**: Reconstruction-based anomaly detection offers three main options for calculating anomaly scores: Point-wise Differencing (PD), Area Differencing (AD), and Dynamic Time Warping (DTW). These methods can be seen as ways to measure the error between the original data and the reconstructed data, but they differ in how they handle that error. Point-wise Differencing (PD) method focuses on the error at each individual data point. It simply takes the absolute difference between the original value and the reconstructed value. This approach is sensitive to even small errors and might be susceptible to noise in the data. The Area Differencing (AD) method looks at a wider window of data points around each point. It calculates the difference between the area under the curve of the original data and the area under the curve of the reconstructed data within a fixed window size. This approach is less sensitive to small, isolated errors but might miss larger trends of discrepancy. Dynamic Time Warping (DTW) method is the most sophisticated and allows for "warping" the time axis. It allows for finding the best possible alignment between the original data and the reconstructed data, even if they are slightly out of sync. This approach is useful for data with local shifts or distortions compared to the reconstruction. In our experiments with the LEAD data, we observe in Table 8 the point wise error type decreases model performance. In general, DTW is better than AD. However, for building 107 we observe AD to perform better than DTW. This can be attributed to the fact that this is a relatively small dataset; thus, its anomaly doesn't involve significant time warping compared to the reconstruction.
- **Error Combination**: In AER for time series anomaly detection, error combination plays a crucial role in determining the final anomaly score. By combining the reconstruction error (from the auto-encoder) and the prediction error (from the LSTM regression), AER aims to capture a more comprehensive picture of potential anomalies. The 'Mult' approach multiplies the reconstruction error and the prediction error. The rationale behind this is that a significant anomaly should cause both errors to be high, and multiplying them amplifies this effect. However, it can be sensitive to outliers in either error. The 'Rec' method solely relies on the reconstruction error for anomaly scoring. The assumption is that anomalies will cause the auto-encoder to struggle with reconstructing the data accurately, leading to a high reconstruction error. Thus, this might miss anomalies that primarily affect predictions. The 'Sum' involves assigning weights to both the reconstruction error and the prediction error and then summing them to obtain the final anomaly score. The weights determine the relative importance of each error in identifying anomalies. In our dataset (LEAD), the 'sum' method

under-performs probably because we had kept relative importance as default, which is 0.5 (equal weightage to both). The 'Rec' combination shows better performance for building 107, suggesting that it has more contextual anomalies, which can be better predicted with the reconstruction-only error. When the mult method is used on it the low value of prediction error sometimes dominates over the reconstruction error, causing the 'rec' to perform better than 'mult'.

8 Future Works

In addressing the challenges observed in both TadGAN and AER, several key areas for future research emerge. TadGAN, as it shows training instability, could benefit from strategies aimed at stabilizing the adversarial training process. This includes exploring alternative network architectures, implementing gradient clipping methods, and adopting curriculum learning strategies to enhance overall robustness. Moreover, the computational demands of TadGAN necessitate investigation into methods to reduce training costs, such as distributed training and model compression, which could significantly enhance its applicability in real-time anomaly detection scenarios. Enhancing TadGAN's ability to capture nuanced temporal relationships within data is crucial, given its observed limitations in handling complex temporal dependencies effectively. Integrating additional mechanisms within the generator architecture, such as attention layers or recurrent neural networks, could address these shortcomings and improve overall performance. Both TadGAN and AER encounter challenges related to false positive rates in anomaly detection. Future research could focus on exploring alternative distance metrics, developing techniques to model short-term data variations more explicitly, or refining anomaly detection mechanisms to mitigate these issues. AER, despite its integration of reconstruction and prediction for anomaly detection, struggles with detecting subtle anomalies, particularly point anomalies, leading to higher false negative rates. Enhancing its sensitivity could involve investigating anomaly scoring mechanisms or optimizing loss functions to penalize false negatives more effectively. Strategies to improve AER's resilience to data quality issues, such as data augmentation techniques or pre-processing methods like data denoising, are essential steps toward mitigating performance fluctuations caused by noisy data inputs. Furthermore, adapting both TadGAN and AER to non-stationary environments—where data characteristics evolve over time—represents a significant future direction. Incorporating online learning strategies or continual adaptation mechanisms could enable these models to adjust dynamically to changing data patterns, thereby enhancing their overall effectiveness in practical applications. These comprehensive approaches not only address current limitations but also pave the way for more robust anomaly detection systems capable of handling diverse and evolving real-world datasets.

9 Conclusions

In this paper, we analyzed the performance of time series models, namely TadGAN and AER for anomaly detection in the energy sector. We analyzed the architecture of these models, and highlighted important points of novelty, which made them stand out in comparison to other deep learning models for anomaly detection. Ten datasets were chosen, namely channel ids P-1,S-1 from spacecraft SMAP, channel id P-15 from MSL in the NASA Space Telemetry Dataset, wind turbines 21, 22 from the SCADA(Supervisory Control and Data Acquisition) 2015 dataset, building ids 334_61, 234_203,36_9688 from the Power Laws Anomaly Detection Dataset, the Machine Temperature Failure Dataset and the building ids 107,108,139 of the Large Energy Anomaly Detection Dataset. We carried out preliminary data-preprocessing to tackle large gaps or few missing values in the datasets, such as filtering or imputation using knowledge of the data behavior, depending on the expanse and number of missing values present. We then carried out Exploratory Data analysis on these datasets, which included Autocorrelation and Partial Autocorrelation Functions, Seasonal Decompositions and Correlation heatmaps, in order to better select model hyperparameters and features for more effective learning of the data patterns as well as for performance improvement. After doing a time series aggregate into equally spaced data points we converted the data into training sequences for the RNNs, by applying a sliding window with controllable length and step size. We tuned several hyperparameters, such as Critic Iterations (specific to TADGAN), reg-ratio(specific to AER) and general ones such as number of epochs, batch size, window size while training the model, both from knowledge of the data as obtained from EDA, as well as observation of better performance. The error function was computed from the original and reconstructed signal by combining a reconstruction error in 'dtw','area' forms and a critic score(In TADGAN) and prediction error(in AER) using a weighted average parameter that could be varied. Anomalies were flagged depending on the magnitude of these error functions at different points, and the best threshold was computed using the Youden J Statistic from ROC-AUC curve. Model performances were compared mainly on the basis of areas under ROC-AUC curves, along with average precision score as well as contextual F1, precision recall and accuracy scores computed through degrees of overlap, in prediction versus ground truth. Finally, we tried improving model performance on some datasets, by noise removal through EMWA, or converting univariate analysis to a multivariate analysis using lagged features(with high PACF) for greater re-emphasis on past trends, on longer stretches of contextual anomalies during model training. After recording the best performance on different

datasets, in mean and standard deviation format, we did a comparative analysis of both models, as well as a study of their drawbacks from an industrial perspective. Finally, we discussed possible future ways of improvement such as incorporation of attention mechanisms in TadGAN architecture or data augmentation while running AER to better detect anomalies in different kinds of data. In conclusion, this study enhanced our understanding of the strengths and weaknesses of TadGAN and AER for anomaly detection in the energy sector. We have gained valuable insights into effective data preprocessing techniques, model architecture, hyperparameter tuning and evaluation metrics. These insights can potentially contribute to future research and development, helping to improve anomaly detection methods in industrial applications for more reliable and efficient energy management systems.

References

[1] Sarah Alnegheimish. Time series anomaly detection — in the era of deep learning | by sarah alnegheimish | data to ai lab | mit | medium. https://medium.com/mit-data-to-ai-lab/time-series-anomaly-detection-in-the-era-of-deep-learning-64b9d2cff6eb, 2024. Accessed: 2024-07-12.

[2] Zahra Zamanzadeh Darban, Geoffrey I Webb, Shirui Pan, Charu C Aggarwal, and Mahsa Salehi. Deep learning for time series anomaly detection: A survey. *arXiv preprint arXiv:2211.05244*, 2022.

[3] Alexander Geiger, Dongyu Liu, Sarah Alnegheimish, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Tadgan: Time series anomaly detection using generative adversarial networks. In *2020 ieee international conference on big data (big data)*, pages 33–43. IEEE, 2020.

[4] Ketan Kumar. Mastering anomaly detection in time series data: Techniques and insights | by ketan kumar | medium. https://medium.com/@ketan31kumar/mastering-anomaly-detection-in-time-series-data-techniques-and-insights-98fbe94c4258, 2024. Accessed: 2024-07-12.

[5] Pankaj Malhotra, Anusha Ramakrishnan, Gaurangi Anand, Lovekesh Vig, Puneet Agarwal, and Gautam Shroff. Lstm-based encoder-decoder for multi-sensor anomaly detection. *arXiv preprint arXiv:1607.00148*, 2016.

[6] Neri Van Otten. Top 8 most useful anomaly detection algorithms for time series. https://spotintelligence.com/2023/03/18/anomaly-detection-for-time-series/, 2024. Accessed: 2024-07-12.

[7] Takaya Saito and Marc Rehmsmeier. The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. *PloS one*, 10(3):e0118432, 2015.

[8] Lawrence Wong, Dongyu Liu, Laure Berti-Equille, Sarah Alnegheimish, and Kalyan Veeramachaneni. Aer: Auto-encoder with regression for time series anomaly detection. In *2022 IEEE International Conference on Big Data (Big Data)*, pages 1152–1161. IEEE, 2022.

[9] Chuxu Zhang, Dongjin Song, Yuncong Chen, Xinyang Feng, Cristian Lumezanu, Wei Cheng, Jingchao Ni, Bo Zong, Haifeng Chen, and Nitesh V Chawla. A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 1409–1416, 2019.

[10] Hang Zhao, Yujing Wang, Juanyong Duan, Congrui Huang, Defu Cao, Yunhai Tong, Bixiong Xu, Jing Bai, Jie Tong, and Qi Zhang. Multivariate time-series anomaly detection via graph attention network. In *2020 IEEE international conference on data mining (ICDM)*, pages 841–850. IEEE, 2020.