xAILAB
EXPLAINABLE MACHINE LEARNING

Reproduction of Selected State-of-the-Art Methods for Anomaly Detection in Time Series Using Generative Adversarial Networks

Master Thesis

Master of Science in Computing in the Humanities

Anastasia Sinitsyna

October 6, 2023

Supervisor:

1st: Prof. Dr. Christian Ledig
2nd: Ines Rieger, M. Sc.

Chair of Explainable Machine Learning
Faculty of Information Systems and Applied Computer Sciences
Otto-Friedrich-University Bamberg

Abstract

Anomaly detection in time series data is a significant application across various domains, such as computer science, medicine, economics, ecology, etc. Various methods exist for detecting anomalies in time series data, including classical statistical approaches, traditional machine learning methods, and deep learning techniques. Many researchers postulate that deep learning approaches, especially reconstruction-based deep learning methods, outperform classical statistical methods and traditional machine learning methods. This study primarily focuses on reconstruction-based methods based on the Generative Adversarial Network (GAN) approach, initially proposed in 2014 for generating synthetic images and has since been adapted by researchers for anomaly detection in time series data.

The first research aim of this study is to replicate the results of state-of-the-art GAN-based methods: TAnoGAN (2020), which utilises the classical GAN approach, and TadGAN (2021), which introduces a novel variation to the classical GAN architecture. These results are compared to the classical statistical ARIMA method to evaluate their performance. The second research aim of the study is to discover how the stationarity characteristics of time series influence the performance of selected state-of-the-art algorithms. The third research aim of the study is to apply selected statistical and GAN-based methods to weather data to determine which approach can better detect meaningful anomalies in unlabelled weather and temperature data in Bamberg.

The study reveal that results comparable to those originally postulated in the initial articles for each state-of-the-art method could be obtained. Additionally, it is discovered that data characteristics could influence the performance of the selected anomaly detection methods. While testing the selected algorithms on unlabelled data, the number of anomalies detected by the selected state-of-the-art method is compared to those identified by the classical statistical ARIMA method. This investigation confirms that GAN-based anomaly detection methods on unlabelled data could potentially identify more meaningful anomalies than traditional statistical methods under certain parameter settings.

Key words: anomaly detection, ARIMA, GAN, time series, univariate, weather

Abstrakt

Die Erkennung von Anomalien in Zeitreihendaten hat große Bedeutung für verschiedene Bereiche wie Computersicherheit, Medizin, Wirtschaft, Ökologie, usw. Es gibt verschiedene Methoden zur Anomalienerkennung in Zeitreihendaten, darunter klassische statistische Methoden, traditionelle Maschinenlernmethoden und Deep Learning-Methoden. Viele Forscher postulieren, dass Deep-Learning-Ansätze, insbesondere Deep-Learning-Methoden auf Rekonstruktionsbasis, klassische statistische Methoden und traditionelle Maschinenlernmethoden übertreffen. Diese Masterarbeit fokussiert in erster Linie auf Methoden, die auf dem Konzept des Generative Adversarial Networks (GAN) basieren. Dieses Konzept wurde ursprünglich im Jahr 2014 für die Erzeugung synthetischer Bilder vorgeschlagen und wurde seitdem von Forschern für die Erkennung von Anomalien in Zeitreihendaten angepasst.

Das erste Forschungsziel dieser Studie besteht darin, die Ergebnisse der modernsten GAN-basierten Methoden zu reproduzieren: TAnoGAN (2020), das den klassischen GAN-Ansatz verwendet, und TadGAN (2021), das eine neue Variante der klassischen GAN-Architektur einführt. Diese Ergebnisse werden mit der klassischen statistischen ARIMA-Methode verglichen, um ihre Performanz zu ermitteln. Das zweite Forschungsziel der Studie besteht darin, herauszufinden, wie die Stationaritätseigenschaften von Signalen die Performanz ausgewählter moderner Algorithmen beeinflussen. Das dritte Forschungsziel der Studie ist die Anwendung ausgewählter statistischer und GAN-basierter Methoden auf Wetterdaten, um festzustellen, welcher Ansatz besser geeignet ist, sinnvolle Anomalien in unmarkierten Wetter- und Temperaturdaten in Bamberg zu erkennen.

In der Studie wurde festgestellt, dass die erzielten Ergebnisse vergleichbar mit den in den Originalartikeln für jede State-of-the-Art-Methode postulierten Ergebnissen sind. Darüber hinaus wurde beobachtet, dass die Eigenschaften der Daten die Leistung der ausgewählten Anomalieerkennungsmethoden beeinflussen können. Bei Tests der ausgewählten Algorithmen an nicht markierten Daten wurde ein Vergleich zwischen der Anzahl der von der ausgewählten State-of-the-Art-Methode erkannten Anomalien und der Anzahl, die durch die klassische ARIMA-Methode identifiziert wurden, durchgeführt. Diese Studie bestätigte, dass GAN-basierte Anomalieerkennungsmethoden unter bestimmten Parametereinstellungen eine höhere Anzahl von Anomalien auf nicht markierten Daten erkennen können als traditionelle statistische Methoden. Dieses Ergebnis legt nahe, dass sowohl klassische als auch ausgewählte GAN-basierte Methoden empfindlich auf die Auswahl der Hyperparameter reagieren.

Schlüsselwörter: anomaly detection, ARIMA, GAN, time series, univariate, weather

Acknowledgements

I am deeply thankful to Prof. Dr. Christian Ledig for guiding me in defining a research topic. I would also like to express my gratitude to my supervisor, Ines Rieger, M.Sc., for her support and guidance during the research process.

Contents

List of Figures

List of Tables

List of Acronyms

1D CNN	One-Dimensional Convolutional Neural Networks
Ac	Accuracy
AdEx	RealAsExchange Signals Collection of NAB Dataset
ADF	Augmented Dickey-Fuller Test
AI	Artificial Intelligence
AIC	Akaike Information Criterion
ANN	Artificial Neural Networks
ARIMA	Autoregressive Integrated Moving Average
ARMA	Autoregressive Moving Average
Art	ArtificialWithAnomaly Signals Collection of NAB Dataset
AWS	RealAWSCloudwatch Signals Collection of NAB Dataset
c	Constant (Test Parameter)
CDC	Open data area of Climate Data Centre
ct	Constant with Trend (Test Parameter)
ctt	Constant with Quadratic Trend(Test Parameter)
DGP	Data Generating Process
DWD	Deutscher Wetterdienst
DS-Stationary	Difference Stationary Time Series
F_1	F_1-Score
FP	False Positive
FN	False Negative
GAN	Generative Adversarial Networks
IQR	Interquartile Range
Known	RealKnownCause Signals Collection of NAB Dataset
KPSS	Kwiatkowski-Phillips-Schmidt-Schin Test
LSTM	Long Short-Term Memory
n	no Constant, no Trend (Test Parameter)
NAB	Numenta Anomaly Benchmark Dataset
Pr	Precision
RNN	Recurrent Neural Network
Re	Recall
SARIMA	Seasonal Auto-Regressive Integrated Moving Average
TadGAN	Time Series Anomaly Detection with GAN
TAnoGAN	Time Series Anomaly Detection with GAN
TN	True Negative
TP	True Positive
Traffic	RealTraffic Signals Collection of NAB Dataset
TS-Stationary	Trend Stationary Time Series
Tweets	RealTweets Signals Collection of NAB Dataset
WD	Weather Dataset

Notation

This section provides a concise reference describing notation as used in the book by Goodfellow et al. (2016).

Numbers and Arrays

a	A scalar (integer or real)
$\boldsymbol{a}$	A vector

Sets and Graphs

$\mathbb{A}$	A set
$\mathbb{R}$	The set of real numbers
$\{0, 1\}$	The set containing 0 and 1
$\{0, 1, \dots, n\}$	The set of all integers between 0 and n
$[a, b]$	The real interval including a and b

Indexing

a_i	Element i of vector $\boldsymbol{a}$, with indexing starting at 1
a_{-i}	All elements of vector $\boldsymbol{a}$ except for element i

Calculus

$\frac{dy}{dx}$	Derivative of y with respect to x
$\frac{\partial y}{\partial x}$	Partial derivative of y with respect to x
$\nabla_{\boldsymbol{x}} y$	Gradient of y with respect to $\boldsymbol{x}$
$\nabla_{\boldsymbol{X}} y$	Matrix derivatives of y with respect to $\boldsymbol{X}$
$\int f(\boldsymbol{x}) d\boldsymbol{x}$	Definite integral over the entire domain of $\boldsymbol{x}$
$\int_{\mathbb{S}} f(\boldsymbol{x}) d\boldsymbol{x}$	Definite integral with respect to $\boldsymbol{x}$ over the set $\mathbb{S}$

Probability and Information Theory

$P(\mathrm{a})$	A probability distribution over a discrete variable
$p(\mathrm{a})$	A probability distribution over a continuous variable, or over a variable whose type has not been specified
$\mathrm{a} \sim P$	Random variable a has distribution P
$\mathbb{E}_{\mathrm{x}\sim P}[f(x)]$ or $\mathbb{E}f(x)$	Expectation of $f(x)$ with respect to $P(\mathrm{x})$
$\mathrm{Var}(f(x))$	Variance of $f(x)$ under $P(\mathrm{x})$
$\mathrm{Cov}(f(x), g(x))$	Covariance of $f(x)$ and $g(x)$ under $P(\mathrm{x})$
$\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$	Gaussian distribution over $\boldsymbol{x}$ with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$

Functions

$f : \mathbb{A} \to \mathbb{B}$	The function f with domain $\mathbb{A}$ and range $\mathbb{B}$
$f \circ g$	Composition of the functions f and g
$f(\boldsymbol{x}; \boldsymbol{\theta})$	A function of $\boldsymbol{x}$ parameterised by $\boldsymbol{\theta}$. (Sometimes we write $f(\boldsymbol{x})$ and omit the argument $\boldsymbol{\theta}$ to lighten notation)
$\log x$	Natural logarithm of x
$\sigma(x)$	Logistic sigmoid, $\dfrac{1}{1+\exp(-x)}$
$\tanh(x)$	Hyperbolic tangent function, $\dfrac{e^x - e^{-x}}{e^x + e^{-x}}$
$\mathbf{1}_{\text{condition}}$	is 1 if the condition is true, 0 otherwise

Datasets and Distributions

p_{data}	The data generating distribution
$\hat{p}_{\text{data}}$	The empirical distribution defined by the training set
$\mathbb{X}$	A set of training examples

1 Introduction

In the modern world, the volume of data collected and analysed by public and private companies is growing daily. Within this landscape, time series data assumes particular importance.

A time series refers to statistical data concerning the values of one or more predefined parameters of a process. These values are collected at regular time intervals, with a fixed gap between each interval Shumway et al. (2000).

Time series are usually periodic and synchronous, which enhances their significance for describing the behaviour of complex systems. Time series used for various tasks, including process modelling, prediction of expected outcomes, classification and clustering Chan and Cryer (2008). Time series forecasting is extensively employed in diverse fields, including disaster prediction, financial crisis prediction, and epidemic projection Laxhammar (1995).

Time series data can include anomalies. Anomalies refer to deviations in data patterns that diverge from the typical behaviour of the series Chandola et al. (2009). Anomalies can significantly affect the accuracy of predictions and also cause false predictions. Anomaly detection has practical applications in various fields. In cybersecurity, unusual network traffic can indicate an attack on a system Alrashdi et al. (2019). Anomalies detected in MRI (Magnetic Resonance Imaging) and ECG (Electrocardiogram) data can serve as indicators of equipment malfunction or potential medical issues in patients, necessitating further diagnostic evaluation Braei and Wagner (2020). An unusually high number of credit card transactions can signify that a card has been compromised or stolen, warranting immediate action such as blocking the card to prevent fraud.

Anomalies can be either point anomalies, where the anomaly is observed in individual time intervals (hereafter measurements or points) or group anomalies, where the anomaly relates to a pattern of a group of data, while the behaviour of individual points in the group remains normal Blázquez-García et al. (2021). Contextual anomalies are another category where a normal data pattern can be identified as anomalous when considered within its context Chandola et al. (2009). An illustrative example is credit card transactions originating from an unfamiliar country for the cardholder.

Indeed, anomaly detection in time series is a challenging task Chandola et al. (2009). Manual anomaly detection is typically feasible in cases of point anomalies, where the behaviour of a single data point within a time series significantly deviates from established thresholds or norms Blázquez-García et al. (2021). Manual detection becomes challenging in more complex scenarios, such as single anomalies with small deviations from normal behaviour or in the case of group anomalies Chandola et al. (2009). All algorithms in anomaly detection can be categorised into three main groups: statistical approaches, which utilise classical statistical models; machine learning approaches, which leverage AI algorithms; and deep learning approaches, encompassing methods that utilise deep learning patterns Schmidl et al. (2022).

The utilisation of traditional statistical algorithms is constrained by their susceptibility to the characteristics inherent in time series data Breiman (2001). Simultaneously, classical machine learning approaches encounter challenges stemming from the absence of definitive *ground truth*, which denotes information regarding categorising time series behaviour as either typical or anomalous for each point in the time series. This limitation reduces the effectiveness of supervised learning methods and makes it necessary to use approaches based on unsupervised learning. In 2014, Goodfellow et al. (2014) introduced a novel image generation method based on game-theory principles, termed Generative Adversarial Network (GAN). Subsequently, this method has been adapted by other researchers for various tasks in time series data Brophy et al. (2023), including prediction in financial markets (TimeGAN Yoon et al. (2019)), imputation of missing values in signals (RGAN Sun et al. (2018)), and anomaly detection (LSTM-GAN Leangarun et al. (2018), MAD-GAN Li et al. (2019), TAnoGAN Bashar and Nayak (2020), and TadGAN Geiger et al. (2020)).

This study is focused on the following research questions:

1. Is it possible to replicate the results of two selected state-of-the-art GAN-based methods for anomaly detection: TAnoGAN Bashar and Nayak (2020), which utilises a classic GAN approach, and TadGAN Geiger et al. (2020) enhance a novel GAN-based method that utilises two generators and two discriminators in interconnected relationships? What are the key differences in terms of accuracy, precision, recall, and F_1-Score between these methods? How is the performance of chosen GAN methods compared with the classical statistical Autoregressive Integrated Moving Average (ARIMA) method?

2. The time series data is analysed concerning different types of stationarity using classical statistical tests: Augmented Dickey-Fuller (ADF) test and Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test. How do the different types of stationarity influence the performance of selected GAN-based methods?

3. The best selected state-of-the-art GAN-based method is used to detect anomalies in an unlabelled Weather dataset for temperature and pressure observations for Bamberg from 1961 to 2020. For comparison, a classical statistical ARIMA method is applied. How does the amount of detected anomalies differ depending on the selected method? Is the number of detected anomalies similar for temperature and pressure modalities in the Weather dataset? Is the number of anomalies in the selected Weather dataset increased for 2000-2020 compared to the observations for 1961-1980?

This thesis is structured as follows: Chapter 2 presents the fundamental theoretical concepts behind the research. In Chapter 3 transitions to examining the core methods employed in this study. Chapter 4 describes the datasets used in this study, offering information about the data and their characteristics. Chapter 5 presents the results of the experiments, which are further discussed in Chapter 6. Chapter 7

encapsulates the research with the conclusion, summarising the research outcomes and outlining future research perspectives.

2 Background

In this chapter, the theoretical background for the study is discussed. Chapter 2.1 provides the theoretical foundation for time series, encompassing definitions, decomposition, and stationarity. Chapter 2.2 introduces the concept of anomaly detection in time series, covering its definition, challenges, types of anomalies related to time series data, and classification approaches for anomaly detection methods. Chapter 2.3 explores different types of neural network architectures, including Artificial Neural Networks (ANN), Recurrent Neural Networks (RNN), Long Short-Term Memory networks (LSTM), and One-Dimensional Convolutional Neural Networks (1D CNN). Chapter 2.4 provides information about Generative Adversarial Networks (GAN).

2.1 Time Series

2.1.1 Definition

A time series can be defined as a sequence of values ordered by time Shumway and Stoffer (2000). Let exists a random set of time points $T = t_1, t_2, \ldots, t_n$, where $n \leq \infty$ occurs. Suppose there is a sequence of observations $X_1, X_2, \ldots$, where X_1 corresponds to the value at the first time point, X_2 to the second, and so on.

Definition 1 (Univariate time series)**.** *A univariate time series $X = \{X_t\}_{t \in T}$ is an ordered sequence of numerical observations, each associated with a specific time moment $t \in T$, where $T \subseteq \mathbb{R}$ Blázquez-García et al. (2021)*

Figure 1 illustrates an example of a univariate time series representing the monthly number of airline passengers in 1949-1960. [1]

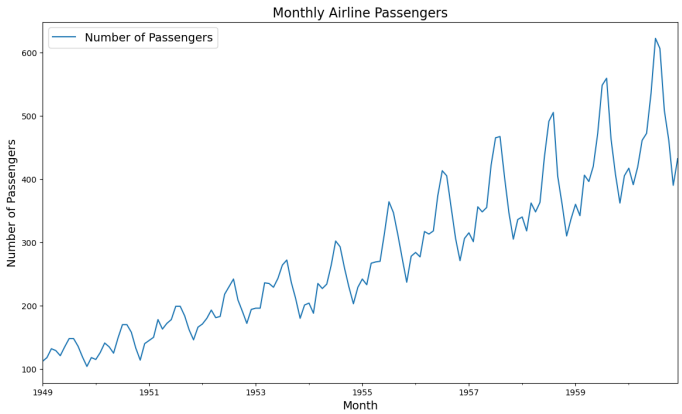


Figure 1: Monthly Airline Passenger Numbers 1949-1960.

Time series can also be distinguished by their origin. There are time series generated from real data, as shown in Figure 1, as well as artificially generated data, like the

[1] https://www.kaggle.com/datasets/rakannimer/air-passengers

white noise time series presented in Figure 2. Artificially generated time series are extensively utilised for testing models across various fields, such as economics, immunology and data science.

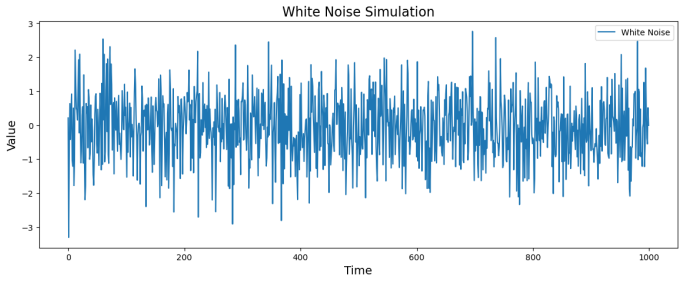


Figure 2: White noise ($WN(0,1)$).

Generation of time series involves the influence of various factors, which can be categorised into the following groups Hamilton (1994):

- **Trend factors:** These factors determine the overall direction of the time series and are typically represented by a monotonic random function $f_{mp}(t)$.
- **Seasonal factors**: These factors give rise to recurrent patterns within the time series, occurring at distinct calendar intervals such as annually, quarterly, monthly, etc. Their effects are characterised by the function denoted as $\phi(t)$.
- **Cyclical factors**: These factors also introduce fluctuations within the series, but these fluctuations are more extended and less consistent compared to seasonal factors. Seasonality alone cannot account for them, and they are described by the function $\psi(t)$.
- **Random factors**: These factors represent the stochastic nature of the time series and are denoted as $\epsilon(t)$. They result from random influences that cannot be predicted in advance.

Factors can affect the generation of a time series in numerous ways, and generally, not all factors are present simultaneously and seasonal and cyclical factors cannot coexist. At the same time, random factors are essential as they reflect the stochastic nature of the time series.

Definition 2. *Each element of the time series can be expressed as follows Hamilton (1994):*

$$X_t = X(t) = \xi(A) \cdot f_{mp}(t) + \xi(B) \cdot \phi(t) + \xi(C) \cdot \psi(t) + \epsilon(t) \tag{1}$$

where (D can be A, B, or C):

$$\xi(D) = \begin{cases} 1, & \text{if factors of type } D \text{ affect the value of } x(t), \\ 0, & \text{otherwise.} \end{cases}$$

.

2.1.2 Time Series Decomposition

Time series decomposition is an analytical process intended to decompose a time series into its fundamental components, such as trend, seasonality, and noise Hamilton (1994). There are several methods of time series decomposition to decompose time series, including additive and multiplicative models. In the additive model, it is assumed that the values of the time series can be decomposed into the sum of a trend, seasonality, and noise, whereas the multiplicative model includes their multiplication. The selection of these methods is contingent upon both the characteristics of the data and the preferences of the researchers. Shumway et al. (2000).

Based on Definition 2, a formal definition of time series decomposition can be derived as follows:

Definition 3 (Time series decomposition)**.** *Decomposition of a time series is the process of splitting each element of the time series into components so that each individual element of the time series can be represented as follows:*

$$X_t = m_t \times S_t \times Y_t \tag{2}$$

where: m_t *refers to the trend component,* S_t *refers to the seasonal component,* Y_t *refers to noise and* $\times = \{+(\textit{additive model}), \times(\textit{multiplicative model})\}$

After the time series decomposition process, each component can be analysed separately, allowing for an understanding of its impact on the overall data dynamics. This analytical process enables forecasting future values and uncovering hidden patterns and trends in the time series data. Figure 3 presents an additive decomposition of the monthly average training signal from the Weather Dataset, which was utilised in this study for anomaly detection.

Trend A trend (m_t) represents a specific, monotonic curve describing the development tendency of a time series. Trends can be either increasing, where the trend values at time $t+1$ exceed the values at time t, or decreasing in the opposite case Hamilton (1994). Time Series in Figure 1 demonstrates an increasing trend.

Depending on the functional form of the trend, linear trends ($m_t = \alpha+\beta t$), quadratic trends ($m_t = \alpha+\beta t+\gamma t^2$), exponential trends ($m_t = \alpha e^{\beta t}$) can be identified Hamilton (1994).

To extract a trend component from a time series, it is possible to use Moving Average (MA) or frequency filtering methods Shumway et al. (2000).

Seasonality Seasonality in data refers to recurring patterns. For example, if a time series exhibits a seasonal component S_t, then for a certain period $n \in \mathbb{N}$, the following holds true: $S(t) = S(t+n)$.

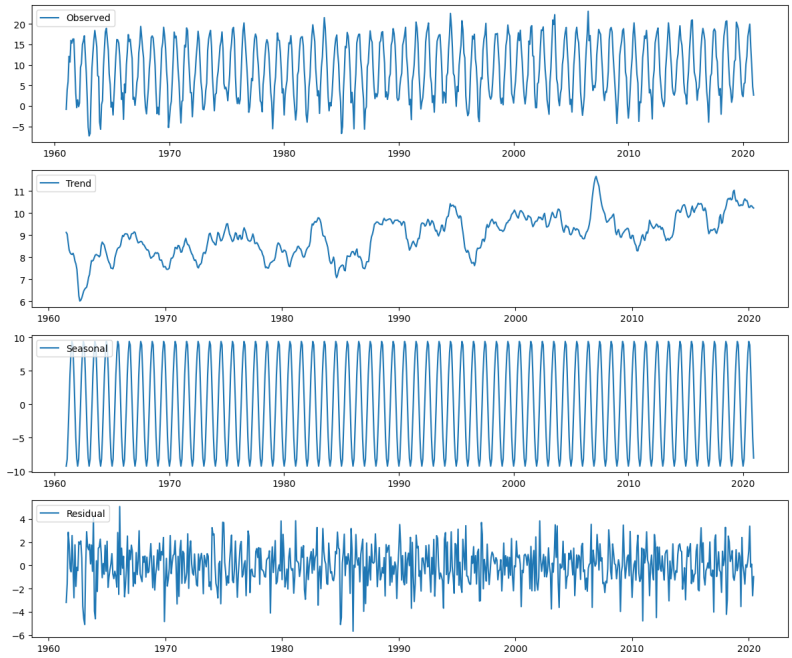


Figure 3: The decomposition plots of monthly average temperature signal in Weather Dataset from 01.1961 to 12.2020

Noise Noise corresponds to irregular, random fluctuations without any explainable correlation with trends, seasonality, or cyclic influences. In classical statistics, white noise (see Definition 8) is employed to model the behaviour of a time series Hamilton (1994).

2.1.3 Stationarity of Time Series

In classical statistics, stationarity of a time series is often considered a necessary condition for the application of certain forecasting models Tsay (1988), such as Autoregressive Moving Average (ARMA) models, because stationary time series have constant statistical characteristics, making them more predictable and allowing for the use of statistical methods and models with greater reliability Hamilton (1994).

A time series $\{X_t\}$ is called strictly stationary if the joint distribution of m observations $X_{t_1}, X_{t_2}, ..., X_{t_m}$ are independent of the time step h, i.e. coincide with the distribution $X_{t_1+h}, X_{t_2+h}, ..., X_{t_m+h}$, for any $h \in N$ Shumway et al. (2000). The prerequisites for strictly stationarity are notably stringent; therefore, in practical terms, weak stationarity is employed for assessing the stationarity of a time series.

Definition 4 (Weakly stationary time series)**.** *A time series $\{X_t\}$ is considered weakly stationary if its statistical properties remain constant over time, namely: $E[X_t] = \mu, \ \forall\ t$, $Var[X_t] = \sigma^2 < \infty, \ \forall\ t$ and the covariance between two time points solely depends on the time difference h: $Cov[X_t, X_{t+h}] = \gamma(h), \ \forall\ t, h$ Shumway et al. (2000)*

The lag variable, denoted by h, represents the time difference between time series elements. The function $\gamma(h)$, dependent on the lag variable, is the time series autocovariance function Hamilton (1994). It is defined for both positive and negative lag values. Due to the symmetry property of covariance, $\gamma(-h) = \gamma(h)$, which makes $\gamma(h)$ an even function. Additionally, for any time points t and s, the relation $cov(x_t, x_s) = \gamma(t - s)$ holds.

Definition 5 (Correlation coefficient)**.** *The correlation coefficient between distinct elements of a stationary time series X_t with lag h is computed as follows:*

$$corr(x_t, x_{t+h}) = \frac{cov(x_t, x_{t+h})}{\sqrt{Var(x_t) \cdot Var(x_{t+h})}} = \frac{\gamma(h)}{\sqrt{\gamma(0) \cdot \gamma(0)}} = \frac{\gamma(h)}{\gamma(0)} \tag{3}$$

Definition 6 (Autocorrelation Function)**.** *The function $p(h) = corr(x_t, x_{t+h})$ is referred to as the autocorrelation function (ACF) of a time series Shumway et al. (2000).*

In addition to the ACF, another important concept is the partial autocorrelation function (PCF), often denoted as $\rho_{\text{part}}(h)$. This function signifies the partial correlation coefficient between time series levels x_t and x_{t+h}, excluding the influence of intermediate levels $x_{t+1}, ..., x_{t+h-1}$.

Definition 7 (Partial Autocorrelation Function)**.** *The partial autocorrelation function (PACF) is: $\rho_{part}(h) = corr(x_t, x_{t+h}|x_{t+1}, ..., x_{t+h-1})$ Shumway et al. (2000)*

A classic illustration of a stationary time series is white noise ϵ_t.

Definition 8 (White noise (WN))**.** *A time series denoted as ϵ_t is considered a white noise process when it represents a random process in which each value of ϵ_t is an independent and identically distributed random variable with a mean $E[\epsilon_t] = 0$ and a constant variance ($Var[\epsilon_t] = \sigma^2$) Hamilton (1994).*

The following models refer to models that are not strictly stationary but can be made stationary through transformations. If a series is trend-stationary (TS-stationary), removing the trend will convert the time series into a stationary one Hamilton (1994).

Definition 9 (TS-Stationary time series)**.** *The time series X_t is called trend stationary (TS) if it can be represented as: $X_t = m(t) + \epsilon_t$, where $m(t)$ is a deterministic function representing the trend or long-term tendency ϵ_t is a stationary time series with a zero mean (white noise) Hamilton (1994).*

In the context of time series analysis, non-stationary time series data can be transformed into stationary using differencing process. This transformation process can be represented by a difference-stationary (DS-Stationary) process denoted as X_t Hamilton (1994). It is expressed as $\Delta^D_{X_t} = \mu + \psi(L)\epsilon_t$, where $\Delta^D = (1 - L)^D$ represents the differencing operator. The term $\psi(L) = \sum_{i=0}^{\infty} \psi_i L^i$ signifies an infinite-degree lag operator polynomial characterised by coefficients that are absolutely summable and all roots located outside the unit circle. Lastly, ϵ_t denotes white noise Box et al. (2015).

2.2 Anomaly Detection

2.2.1 Definition

The definition of the concept of anomaly depends on the specific context of the problem. Anomaly, in general, can be characterised as an observation that stands out due to its distinctiveness compared to the remaining dataset, which has been defined as normal Hawkins (1980).

Definition 10 (Anomalies)**.** *Anomalies in time series, commonly referred to as outliers Tsay (1988), are points or sequences of points in a time series exhibiting behaviour that significantly diverges from the typical patterns observed in the time series Chandola et al. (2009).*

However, this definition requires a precise definition of what constitutes typical patterns. Therefore, Chandola et al. (2009) provide that data that deviate from the expected behaviour of a time series predicted from the analysis of the patterns of past behaviour of the series should be considered anomalies.

In the context of temporal sequences, an anomaly could indicate that the system behaves atypically Geiger et al. (2020) or that the underlying process dynamics are changing. Anomalies stem from errors in data collection or preprocessing external influences Li et al. (2019) (such as abnormal credit card transactions) or can even signify issues beyond the system framework (e.g., climate changes or anomalous medical readings like ECG results).

Formally, the anomaly detection task can be defined as the search for unusual data points that deviate from the overall structure. This is accomplished by comparing new observations to a dataset that is considered normal. If a new observation significantly differs from the normal data, it is assigned an *anomaly* label based on a pre-established threshold value for anomaly scores Tsay (1988).

2.2.2 Challenges in Anomaly Detection on Time Series

Anomaly detection in univariate time series poses significant challenges, primarily due to the following reasons Chandola et al. (2009):

1. **Lack of labelled data**: The absence of labelled data at the initial stages of anomaly detection lacks stringent guidelines, imposing substantial constraints on the method's formalisation.

2. **Stochastic noise**: Stochastic noise within a time series exacerbates the challenge of distinguishing anomalous signals since noise constitutes an inherently normal component of a time series (see Chapter 2.1.1). Algorithms must be capable of discriminating between true anomalies and noise.

3. **Analysis of clustered anomalies**: Determining the exact length of anomalous clusters in group anomaly analyses is difficult. It causes the necessity for the utilisation of algorithms with variable detection dynamics, such as window-based algorithms with variable lengths of windows.

4. **Appropriate metric selection**: The selection of appropriate metrics for anomaly evaluation depends on the specificity of the time series and presents a complex research task. More intricate metrics like Dynamic Time Warping (DTW) might be required for detecting group anomalies, as classical metrics like Euclidean distance are sensitive to thresholds Geiger et al. (2020).

2.2.3 Types of Anomalies

Depending on the scope and data patterns, the following types of anomalies can be identified Tsay (1988):

1. **Point anomalies**: In this case, anomalies in the data occur only at specific points, i.e., the length of the anomalous period is one Blázquez-García et al. (2021). Such anomalies can arise in various data types, such as videos, images, etc. Figure 4 shows a time series with two point anomalies.

2. **Contextual anomalies**: In this scenario, the anomaly is related to external data rather than the characteristics of the time series. The context here includes contextual attributes that define the data environment and behavioural attributes that define the nature of instances Chandola et al. (2009). An example of a contextual anomaly could be negative temperature during the summer period Blázquez-García et al. (2021),Chandola et al. (2009), as is shown in Figure 5, where the same temperature $t_1 = t_2 = 35F$ could be normal during the winter month (t_1) but anomalous if occurs during summer (t_2).

3. **Group anomalies**: In this case, an anomaly arises within a group of points, while each individual point is considered normal. Such anomalies can occur only in datasets where instances are interrelated, such as time series data Chandola et al. (2009). Figure 6 illustrates an anomaly in which the frequency of a time series changes, even though the behaviour of individual points in the series remains within the normal range. This anomaly may indicate failures in the operation of equipment that generates this signal, such as sensor malfunctions.

2.2.4 Classification of Anomaly Detection Approach by Test Data Availability

A mathematical or statistical model that explains the system's behaviour Chandola et al. (2009) and a data set that illustrates the system's functioning Blázquez-García

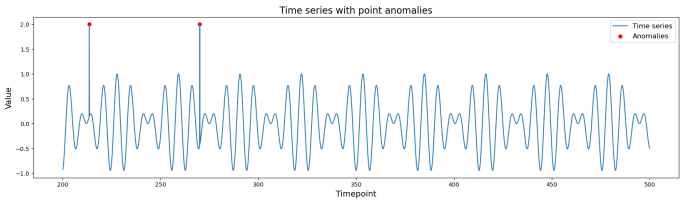


Figure 4: Point anomaly in time series. Source: Synthetic time series generated from $X_t = 0.5 \cdot sin(t) + 0.5 \cdot cos(0.8t)$ for $t_{1,...,100000} \in [1, 1000]$ with point anomalies, that are generated using a random() function from NumPy library.

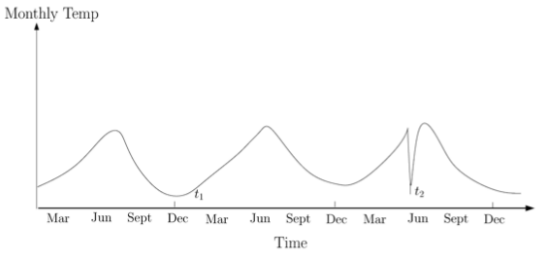


Figure 5: Contextual anomaly t_2 in a temperature time series. Source: Chandola et al. (2009)

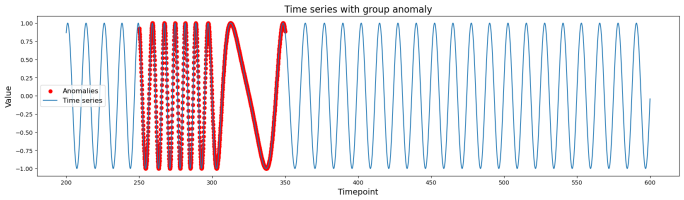


Figure 6: Group anomaly in time series. Synthetic time series generated from $X_t = cos(0.5t \cdot (1 + 0,1 \cdot exp\{-\frac{(t-300)^2}{100}\}))$ for $t_{1,...,100000} \in [1, 1000]$

et al. (2021) are required to identify anomalies. Each element of this set contains a label indicating whether it is normal or anomalous. This allows grouping instances with the same labels into corresponding classes. Creating such a labelled dataset typically demands significant time and resources Chandola et al. (2009). Sometimes, obtaining data for the anomalous class is challenging due to a lack of information about outliers in the system. It is important to note that most time series data are essentially unlabelled Chandola et al. (2009).

Depending on the availability of information regarding data normality or anomaly, Chandola et al. (2009) propose categorising anomaly detection methods into the following groups:

1. **Supervised learning**: This method requires a training sample that must fully represent the system and include both normal and anomalous data. During the training process, a model is constructed for both the normal and abnormal data classes. In the testing process, each observation (data sample, a point

in case of a time series) is compared to each of the classes and a decision is made on which class the observation belongs to. Challenges of this method include class distribution imbalance, as the anomalies are significantly fewer in datasets compared to observations with normal values. Another challenge is the complexity of obtaining anomalous labels in time series Geiger et al. (2020).

2. **Semi-supervised learning**: In this method, the training utilises a dataset with known labels for the normal class. Similarly, during testing, it is determined whether an observation belongs to the normal class or not.

3. **Unsupervised learning**: Unsupervised anomaly detection is a data analysis method aimed at detecting anomalies without using predefined labels. In this approach, the model is trained on unlabelled data, enabling it to capture typical patterns and structures in the data. Then, by analyzing new data, the model identifies anomalies based on deviations from learned patterns. This approach assumes that the majority of data belongs to the normal class. Otherwise, it is possible that the method may classify anomalies as normal data Blázquez-García et al. (2021). The major advantage of this method is its ability to detect unknown anomalies that were not foreseen in the training sample.

For detecting anomalies in time series data, unsupervised methods are more suitable Geiger et al. (2020).

2.2.5 Classification of Anomaly Detection Approaches by Method of Origin

A common approach involves classifying the techniques used for anomaly detection in time series into the following groups. Statistical methods are based on the assumption that data has been generated using a specific statistical model Breiman (2001). The classical machine learning approach operates under the assumption that information about the data generation process is not available (referred to as the *black box* approach) and that all assumptions must be derived based only on the available data Breiman (2001). Deep neural network-based methods rely on some type of deep neural network Schmidl et al. (2022).

Statistical methods are methods whose fundamental principles were developed in the 20th century. These methods encompass various techniques, including decomposition methods based on identifying unusual data patterns that can be interpreted as anomalies. For example, Principal Component Analysis (PCA) Crépey et al. (2022) is used to analyse raw data and extract principal components that contain the most information. Anomalies can be defined as data points that deviate from these principal components. Prediction based methods are used to predict future time series values based on time series data and statistical models behind the data

generating process. These methods employ the historical values of a time series $\{X_t\}$ to predict the value of X_{t+1} Tsay (1988). These methods help analyse and forecast trends, patterns, and changes in the data.

Among these methods, methods based on statistical models could be distinguished. This includes ARMA Hamilton (1994), ARIMA Box et al. (2015), Kalman filters Ting et al. (2007). The ARMA and ARIMA methods will be described in detail in Chapter 3.2.

The anomalous score for these methods refers to the difference between the predicted $\hat{X}_{t+1}$ value and the actual X_{t+1} value. Statistical methods are sensitive to their parameters Shumway et al. (2000), and selecting appropriate parameters can be challenging. Incorrect parameter choices can lead to inaccurate predictions and suboptimal performance Alabdulrazzaq et al. (2021). On the positive side, these methods are straightforward and do not require significant computational resources.

Classical Machine Learning Methods These methods utilise machine learning but do not rely on neural network-based approaches. In this category, three subgroups can be identified, as discussed in Domingues et al. (2018):

Isolation Methods: These methods classify a data point as an anomaly if it significantly differs from most of the data. Prominent examples include the Isolation Forest and the Hybrid Isolation Forest. These techniques are particularly effective at detecting isolated point anomalies.

Neighbourhood-Based Methods: Methods in this category analyse the local context of each data point to identify outliers. They may utilise metrics such as the distance to the k-nearest neighbour, as demonstrated by the Local Outlier Factor (LOF) Breunig et al. (2000). Alternatively, they may employ clustering techniques to identify high-density regions, considering points outside of these regions as anomalies, as shown in the case of the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) Wang et al. (2015).

Deep Neural Networks Methods This category encompasses all methods that are fundamentally based on deep neural networks. In recent years, these methods have experienced significant advancements, as substantiated by research studies Domingues et al. (2018), Darban et al. (2022).

Many techniques that utilise deep neural networks can be categorised as reconstruction methods. These methods have the potential advantage of gathering information from data and learning without the need for ground truth, especially when compared to classical machine learning methods. For this reason, in the case of unlabelled time series data, deep learning methods appear potentially attractive for addressing anomaly detection problems in signals.

2.3 Neural Networks

2.3.1 Artificial Neural Network

A neural network is a system of interconnected neurons that exchange information through synapses. Artificial neural networks (ANN) aim to mimic the functioning of the human nervous system and have applications in various domains, including speech recognition, object recognition Abiodun et al. (2018). ANN commonly handle classification Abid and Hamami (2018), value prediction, and optimisation tasks Yu et al. (2009).

In its simplest form, a neural network consists of layers of neurons connected by synapses Abiodun et al. (2018). Neurons receive, process, and transmit information within the network. Neurons are organised into input, hidden, and output layers (see Figure 7). Input data flows through the layers, with each neuron's input being the sum of inputs from the previous layer, normalised through an activation function. Synapses have weights that influence information transmission. The final layer of the network generates the system's output signal.

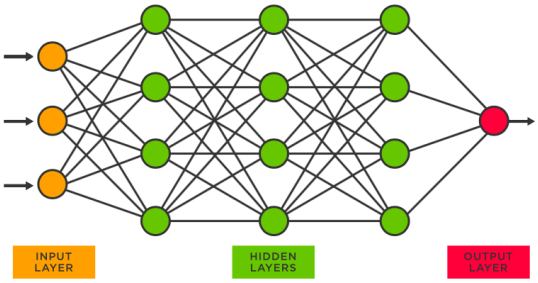


Figure 7: Neural Network with an input layer (red), three hidden layers (green) and one output layer (red). Sourse: `https://www.tibco.com/reference-center/what-is-a-neural-network`

2.3.2 Recurrent Neural Network

A Recurrent Neural Network (RNN) (see Figure 8) is a neural network with internal memory designed for sequential data. One of the defining features of an RNN is its ability to process data sequences, rendering it well-suited for tasks like speech recognition and handwriting recognition. In an RNN, a consistent operation is performed for each element within a sequence. The output hinges on the outcomes of preceding computations, creating a loop where the output becomes input for the next steps. This way, the RNN considers both the current input and information from previous steps when making decisions.

Unlike feed-forward ANN where inputs are treated as independent entities, RNN interconnect all inputs.

RNNs are great at maintaining context during training and can connect previous information to the current task. However, they can struggle with long-term dependencies, making precise parameter tuning challenging.

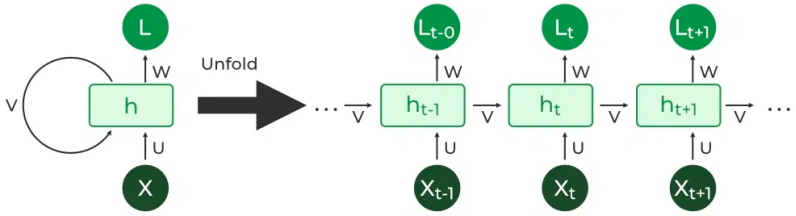


Figure 8: A recurrent neural network (RNN): On the left side is presented a loop configuration that connects different elements. On the right side, a simple, direct view is presented. Sourse: https://www.geeksforgeeks.org/introduction-to-recurrent-neural-network/

Unlike feed-forward ANN where inputs are treated as independent entities, RNN interconnect all inputs. An appealing aspect of RNN is their potential to establish connections between previous information and the ongoing task

Despite RNN theoretically possessing the capacity to manage long-term dependencies, they often struggle to utilise such dependencies in practical applications effectively. Although humans can meticulously fine-tune network parameters to solve artificial problems involving long-term dependencies, training RNN with such precision remains challenging Hochreiter and Schmidhuber (1996).

2.3.3 Long Short-Term Memory

Long Short-Term Memory (LSTM) represents a distinctive variant of recurrent neural network architecture designed specifically for capturing long-term dependencies within data. The primary objective of LSTM is to tackle the challenge of learning and effectively utilising information across extended time intervals.

The structure of LSTM (see Figure 9) resembles a chain, similar to conventional recurrent networks, but it boasts a more intricate organisation.

LSTM employs three gate modules: forget, input, and output, to control information flow. These gates determine what to keep, what to use in the current step, and what to pass to the next step. A central feature of LSTM is the memory cell, which stores and preserves information. The cell can be updated through the gate modules, using activation functions like the hyperbolic tangent to manage data flow.

Crucially, LSTM includes a cell state, a horizontal line atop the architecture. Information can be selectively removed from the cell state via gates. These gates consist of sigmoidal layers and component-wise multiplications. The sigmoidal layer decides how much information should pass, generating values between 0 and 1. Each LSTM cell has three such gating mechanisms.

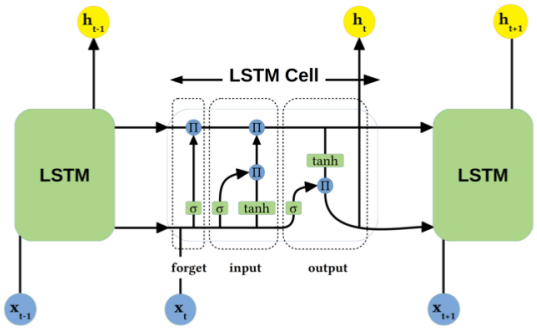


Figure 9: Long Short-Term Memory Architecture. Source: Schmidl et al. (2022)

.

The algorithm within an LSTM cell unfolds as follows Schmidl et al. (2022):

1. **Forgetting the Irrelevant Information**: Using the "forget layer," the algorithm decides what information from the previous cell state C_{t-1} to retain or discard based on h_{t-1} and x_t. It assigns values between 0 and 1: $f_t = \sigma(W_f[h_{t-1}, x_t] + b_f)$

2. **Incorporating New Information**: This step is divided into two stages. Initially, the "input layer" identifies which values need updating:

   $i_t = \sigma(W_i[h_{t-1}, x_t] + b_i)$

   Subsequently, the tanh layer constructs a vector of fresh candidate values, $\tilde{C}_t$, to be added to the cell state: $\tilde{C}t = \tanh(W_C[ht-1, x_t] + b_C)$

3. **Updating the Cell State**: The cell state transitions from the previous state C_{t-1} to the new state C_t: $C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$

4. **Generating Output**: This stage employs a combination of sigmoidal and tanh layers to determine the model's output values. Initially, a sigmoidal layer acts as an intermediary, determining which information from the cell state will be conveyed to the output: $o_t = \sigma(W_o[h_{t-1}, x_t] + b_o)$

   Subsequently, the cell state values undergo processing through the tanh layer, normalising them within the -1 to 1 range to mitigate issues related to exploding gradients. These normalised values are then multiplied by the output values of the sigmoidal layer. This mechanism empowers the control over the information transmitted to the output: $h_t = o_t * \tanh(C_t)$

2.3.4 One-Dimensional Convolutional Neural Network

Convolutional Neural Network (CNN) refers to ANN that use alternating layers of convolution and sub-sampling Kiranyaz et al. (2021).

CNN are commonly associated with two-dimensional image classification, where the convolution process considers both width and height. One-dimensional CNNs (1D CNN) are used for analysing temporal data like sensor signals and text. They excel at recognising patterns and dependencies in sequences. In 1D CNN, the convolutional kernel moves along the temporal axis, performing scalar operations Kiranyaz et al. (2021).

1D CNN take raw one-dimensional signals as input. The primary configurations during the setup of 1D CNN encompass the amount of hidden layers/neurons, the filter (kernel) dimensions in each CNN layer, the sub-sampling coefficient in each CNN layer, and the activation function Kiranyaz et al. (2021).

At each layer, a convolution operation involves the kernel traversing the input data, followed by an activation function (most commonly ReLU) imbuing the process with non-linearity. Figure 10 illustrates a simplified 1D CNN structure.

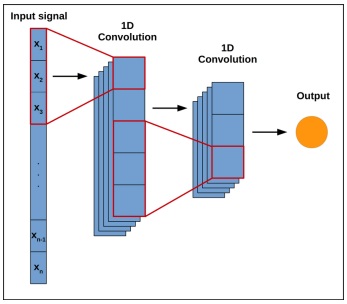


Figure 10: 1D Convolutional Neural Network (CNN) architecture consisting of two convolutional layers. Source: Shenfield and Howarth (2020)

.

2.4 Generative Adversarial Networks

Generative adversarial networks (GANs) originated from Ian Goodfellow Goodfellow et al. (2014) in 2014 as a solution to a class of problems related to generative modelling. The main purpose of generative modelling lies in exploring training examples and the probability distributions behind them, in order to further generate new examples from an assumed probability distribution. GANs were originally designed for image generation Goodfellow et al. (2014), but the game theory principles on which they are based allow them to be applied to other classes of problems, including time series generation Gui et al. (2021).

GANs are based on generative modelling. Regarding generative modelling, training samples, denoted as $\mathbf{x}$, are drawn from an unknown distribution $p_{data}(\mathbf{x})$. The main goal of the method is to construct a model $p_{model}(\mathbf{x})$ that comes as close as possible to the distribution $p_{data}(\mathbf{x})$. A classical statistical approach suggests using the function $p_{model}(\mathbf{x}; \Theta)$ and adjusting the parameter Θ to minimise the discrepancy between p_{model} and p_{data} Goodfellow et al. (2020). This method is well applicable in classical statistics for simple probability distribution functions and small amount

of data in datasets Goodfellow et al. (2020). However, for large real-world datasets and complex distributions, this method becomes inefficient due to the computational complexity of probability density functions Goodfellow et al. (2014).

GANs provide an alternative approach, avoiding the necessity of direct modelling of probability density functions. Instead, the focus is shifted towards the data generation process itself and the functions that generate this data Goodfellow et al. (2020). The architecture of GAN comprises interconnected machine learning models, often implemented as neural networks. The initial network, referred to as the generator $\mathcal{G}$, establishes the model $p_{model}(x)$. The generator does not required to calculate the probability density function p_{model}. Its primary objective is to produce samples from the distribution p_{model}. To achieve this, the generator takes an input vector $\mathbf{z}$ (typically representing white noise) from the latent space $\mathbb{Z}$. The generative function $\mathcal{G}(\mathbf{z}; \Theta^{(\mathcal{G})})$ then transforms this input into realistic data, with $\Theta^{(\mathcal{G})}$ denoting the set of trainable parameters that define the approach of generator for generating realistic data. The role of the second network, known as the discriminator $\mathcal{D}$, is to evaluate the generated samples $\mathbf{x}$ and assign a score $\mathcal{D}(\mathbf{x}; \theta^{(\mathcal{D})})$. This score serves as an indication of whether the sample $\mathbf{x}$ belongs to the training distribution (considered real) or to p_{model}, indicating that the data was generated using the generative function $\mathcal{D}$ from the latent space $\mathbb{Z}$.

For each player, the following cost functions are defined: $J^{(\mathcal{G})}(\theta^{(\mathcal{G})}, \theta^{(\mathcal{D})})$ for the generator and $J^{(\mathcal{D})}(\theta^{(\mathcal{G})}, \theta^{(\mathcal{D})})$ for the discriminator Goodfellow et al. (2020). Each player aims to minimise its cost functions, i.e. the discriminator aims to classify the data accurately and the generator aims to generate samples that the discriminator misclassifies as true. Goodfellow et al. (2020) define two approaches to determining the cost functions of generator. Minmax GAN (M-GAN) defines the generator cost function as the opposite of the discriminator cost function $J^{(G)} = -J^{(D)}$, non-saturating GAN (N-GAN), where the generator's task is to minimise the negative log-likelihood of the assumption that the discriminator misclassified the data.

The classical GAN learning process proposed by Goodfellow et al. (2014) consists of the following steps:

1. Generate m samples $\{\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(m)}\}$ from the distribution $p_{model}(\mathbf{z})$.

2. Obtain m samples $\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}\}$ from the distribution $p_{data}(\mathbf{x})$, representing real data.

3. Randomly shuffle the samples and feed them as input to the discriminator D. The discriminator's output is a value between 0 and 1, where 0 indicates fake data and 1 indicates real data.

4. Calculate the discriminator loss function:

$$\mathcal{D}_{loss} = -\frac{1}{m}\sum_{i=1}^{m}[\log \mathcal{D}(\mathbf{x}^{(i)}) + \log(1 - \mathcal{D}(\mathcal{G}(\mathbf{z}^{(i)})))] \tag{4}$$

   Update the discriminator parameters using stochastic gradient descent: $\nabla_{\theta_d}\mathcal{D}_{loss}$.

5. Repeat steps $1-4$ k times. Parameter k is predefined and often $k=1$ was used to reduce computational costs.

6. Generate m new samples $\{\mathbf{z}^{(1)},\ldots,\mathbf{z}^{(m)}\}$ from the distribution $p_{model}(\mathbf{z})$.

7. Calculate the generator loss function:

$$\mathcal{G}_{loss} = -\frac{1}{m}\sum_{i=1}^{m}\log(1-\mathcal{D}(\mathcal{G}(\mathbf{z}^{(i)}))) \tag{5}$$

   Update the generator using stochastic gradient descent: $\nabla_{\theta_g}\mathcal{G}_{loss}$.

8. Repeat steps $1-7$ for a predetermined number of iterations (*epochs*).

Figure 11 shows the main components of the GAN. The dotted lines denote the interaction processes between the components of the network.

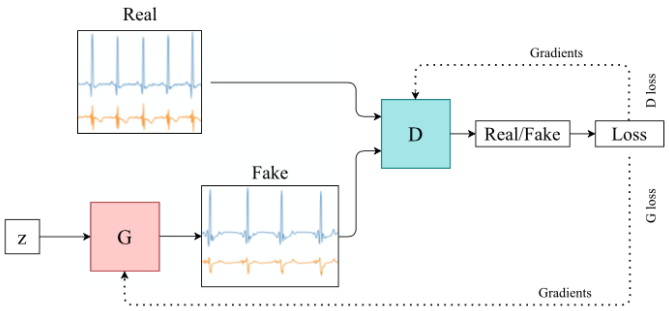


Figure 11: Illustration of the basic modules in the Generative Adversarial Network (GAN) and the relationships between them, where $\mathcal{G}$ represents the generator, $\mathcal{D}$ refers to the discriminator, and $\mathbb{Z}$ represents the latent space used for generating fake data. Source: Brophy et al. (2023)

.

In other words, the objective of $\mathcal{D}$ is to correctly classify incoming samples, while the goal of G is to minimise the function $log(1–\mathcal{D}(\mathcal{G}(\mathbf{z})))$. Consequently, the equation for the classical minimax game between $\mathcal{G}$ and $\mathcal{D}$ can be expressed as follows:

Maximise for $\mathcal{D}$: $\mathbb{E}_{\mathbf{x}\sim p_{\text{data}}}[\log(\mathcal{D}(\mathbf{x}))]$

Minimise for $\mathcal{G}$: $\mathbb{E}_{\mathbf{z}\sim p_z}[\log(1-\mathcal{D}(\mathcal{G}(\mathbf{z})))]$

After combining both parts:

$$min_{\mathcal{G}}max_{\mathcal{D}}V(\mathcal{D},\mathcal{G}) = \mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}[log\mathcal{D}(\mathbf{x}] + \mathbb{E}_{\mathbf{z}\sim p_{\mathbf{z}}(\mathbf{z})}[log(1-\mathcal{D}(\mathcal{G}(\mathbf{x})))] \tag{6}$$

Thus, GANs represent an innovative approach to generative modeling based on game theory principles, enabling the generation of new data closely resembling the original distribution while avoiding complex computations of probability density functions. In this thesis, two GAN-based approaches were selected to evaluate the performance of anomaly detection methods based on GAN-based time series reconstruction for univariate time series.

3 Methods

This chapter outlines the methods used to answer the research questions in this study. Chapter 3.1 provides information about the various methods used for the investigation of signal stationarity. Chapter 3.2 describes the chosen baseline ARIMA method, and Chapter 3.3 represents selected state-of-the-art GAN-based methods. Chapter 3.4 covers the description of evaluation metrics for assessing method performance.

3.1 Stationarity Tests

One approach to testing time series stationarity involves graphical techniques such as plotting. These graphical methods include correlation plots in the form of autocorrelation (ACF) plots and partial autocorrelation (PACF) plots. These plots depict correlations between data points at different time lags and include a confidence interval for significance determination.

In stationary time series, ACF plots typically display a rapid correlation decay with increasing lag, regardless of the initial lag's value. Conversely, non-stationary series often start with a high correlation (ACF(1) close to 1) and gradually decline with higher lags. PACF plots offer similar insights. Stationary series commonly show a swift correlation drop after initial lags, while non-stationary series may have a high PACF(1) value but insignificance in later coefficients within the confidence interval.

Other verification methods involve applying statistical tests for stationarity. Prominent examples include the Augmented Dickey-Fuller (ADF) Dickey and Fuller (1979) Test and the Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test Kwiatkowski et al. (1992). ADF test inspects the DS-Stationarity approach, and KPSS test focuses on TS-Stationarity in time series.

3.1.1 Augmented Dickey-Fuller (ADF) Test

The Augmented Dickey-Fuller (ADF) test focuses on testing the hypothesis of the existence of a unit root in a time series. If a unit root is present, the time series is considered non-stationary. Assume, that a time series X_t exists. The ADF model takes the following form:

$$\Delta X_t = \alpha + \beta t + \gamma X_{t-1} + \delta_1 \Delta X_{t-1} + \cdots + \delta_{p-1} \Delta X_{t-p+1} + \epsilon_t \tag{7}$$

where ΔX_t refers to the first difference of the time series at time t, α is the constant term, β is the coefficient in front of the time t, which tests for statistical significance, γ is the coefficient in front of the lagged value of the series, $\delta_1, \delta_2, \ldots, \delta_{p-1}$ are coefficients in front of the first time series differences at previous lags, p refers to lag order, ϵ_t is the random error term at time t Hamilton (1994).

Null hypothesis H_0:$\gamma = 0$

Alternative hypothesis H_1:$\gamma < 0$

After calculating the test statistic $DF_\tau = \frac{\hat{\gamma}}{\text{SD}(\hat{\gamma})}$, it can be compared to the corresponding critical value specific to the ADF test. Because of the asymmetry of this test, the focus lies solely on the negative values of the test statistic. Should the calculated test statistic be smaller than the critical value, it leads to the rejection of the null hypothesis Dickey and Fuller (1979).

3.1.2 Kwiatkowski-Phillips-Schmidt-Shin (KPSS) Test

The Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test is a statistical test used to check whether a time series is stationary with respect to the trend (see Definition 9) The KPSS test has null and alternative hypotheses as follows:

Null hypothesis H_0**:** The time series is stationary around the trend.

Alternative hypothesis H_1**:** The time series is non-stationary or has a unit root.

The main idea behind the KPSS test is to analyse how the series deviates from stationarity Kwiatkowski et al. (1992). It is based on the following model of a time series:

$$X_t = m_t + V_t \tag{8}$$

where: X_t is the value of the time series at time t, m_t is the trend component of the time series, V_t is the random component of the time series.

If the test statistic value exceeds the critical values, the null hypothesis is rejected, indicating that the series is considered non-stationary. Conversely, if the test statistic value is smaller than the critical values, the null hypothesis is not rejected, suggesting that the series is considered to be TS-Stationary Harvie and Pahlavani (2006).

3.2 Baseline Method for Anomaly Detection: Autoregressive Integrated Moving Average (ARIMA)

Autoregressive Integrated Moving Average (ARIMA) is a statistical methodology introduced in the 1970s by George Box and Gwilym Jenkins Box et al. (2015) for analysing and forecasting time series data. The model is characterised as *ARIMA(p, d, q)*, where p denotes the amount of autoregressive lags (or autoregressive order), d signifies the degree of integration, and q represents the moving average order Hamilton (1994). The ARIMA model for a non-stationary time series X_t in its general form can be expressed as Shumway et al. (2000):

$$\triangle^d X_t = c + \sum_{i=1}^{p} \alpha_i \triangle^d X_{t-i} + \sum_{j=1}^{q} \beta_j \epsilon_{t-j} + \epsilon_t \tag{9}$$

Where:

- ϵ_t is a stationary time series (typically $\epsilon \sim WN(0, 1)$).

- c, is constant and α_i for $i \in \{1, \ldots, p\}$, β_j for $j \in \{1, \ldots, q\}$ represent the model parameters.
- $\triangle^d$ is the differencing operator applied to the time series.

ARIMA main components are Autoregressive (AR), Integrated (I), and Moving Average (MA) Abudu et al. (2010). To attain stationarity, the ARIMA model employs differencing, which involves calculating the difference between successive values of the original series—referred to as transitioning to a sequence of increments:

$$\triangle^1 X_t = X_t - X_{t-1}$$
$$\triangle^2 X_t = \triangle X_t - \triangle X_{t-1}$$
$$\ldots$$
$$\triangle^d X_t = \triangle^{d-1} X_t - \triangle^{d-1} X_{t-1}$$

$$\triangle^d X_t \sim ARMA(p, q) \rightarrow X_t \sim ARIMA(p, d, q)$$

If a sequence of increments of order d ($\triangle^d X_t^d X_t$) exhibits stationarity, then the original time series X_t is said to be integrated of order d. This process eliminates trends and seasonal variations that render the series non-stationary. The necessary differencing order d is determined empirically by studying the series graph. Pronounced shifts in the graph's level or fluctuations call for first-order differencing, whereas changes in slope (trend) necessitate second-order differencing.

The Autoregressive Moving Average ($ARMA(p, q)$) refers to a forecasting model for stationary series and is defined as:

$$X_t = c + \epsilon_t + \sum_{i=1}^{p} \alpha_i X_{t-i} + \sum_{i=1}^{q} \beta_i \epsilon_{t-i}$$

, with c as a constant, ϵ_t as white noise, $\alpha_1, ..., \alpha_p$ as auto regressive coefficients, and $\beta_1, ..., \beta_q$ as moving average coefficients.

ARMA process consists of two processes: Moving Average (MA) and AutoRegressive (AR).

Definition 11 (Moving Average)**.** *The Moving Average of order q ($MA(q)$) is a time series model given by:*

$$X_t = \sum_{j=1}^{q} \beta_j \epsilon_{t-j} \tag{10}$$

where ϵ_t representing white noise and β_j for $j \in \{1, \ldots, q\}$ represent the model parameters.

The MA model presupposes that the current value of the time series is linearly dependent on the previous forecast residuals (errors). The parameters $\beta_1, ..., \beta_q$ determine the influence of prior residuals on the current value.

Definition 12 (Autoregression)**.** *The Autoregression of order p (AR(p)) is a time series model wherein a moment's values depend linearly on preceding values of the same series. Mathematically, the AR(p) process can be expressed as:*

$$X_t = c + \epsilon_t + \sum_{i=1}^{p} \alpha_i X_{t-i} \tag{11}$$

with $\alpha_1, ..., \alpha_p$ as model parameters, c as a constant (often assumed as zero for simplicity), and $\epsilon_t \sim WN(0, 1)$.

Parameters of $ARIMA(p, d, q)$ morel should be empirically defined, using statistical tests Shumway et al. (2000), Hamilton (1994). The selection of appropriate parameters can be challenging and strongly influences the accuracy of the prediction of the chosen model Shumway et al. (2000).

Seasonal-ARIMA If the time series exhibits a significant seasonal component, it would be better to use an ARIMA modification designed for seasonal data, such as $SARIMA(p, d, q)(P, D, Q)_s$, where s refers to amount of observations in a year which takes into account the seasonality of the time series.

Auto ARIMA During this study, an automated ARIMA model, *auto.Arima()*, from the *pmdarima* library[2] was used to select the parameters p, q, d, as well as additional P, Q, D, and s for time series with an additional seasonal component.

The fitting process of the ARIMA model can be described as follows:

1. Defining a differencing order d was necessary to transform the original time series into stationary. This was achieved by applying one of two stationarity tests: ADF or KPSS.

2. The selection of different ARMA models (p, q), where $p, q \in \{0, \ldots, \max\}$, and max is defined as the maximum value for p and q coefficients. For each model, the Akaike Information Criterion (AIC) was computed. The model selection process followed this sequence: AIC calculations were initially performed for the following models: ARMA(0,0), ARMA(0,1), ARMA(1,0), and ARMA(2,2). Subsequently, the parameters were adjusted by ± 1, and AIC values were recalculated for the new models. This process continued until parameter adjustments led to a reduction in the AIC.

3. The selection of seasonal parameters was conducted in a similar manner.

[2] `https://pypi.org/project/pmdarima/`

Akaike Information Criterion (AIC) The Akaike Information Criterion is a tool that assists in identifying the most suitable statistical model. It was developed in 1971 by H. Akaike and published in 1974 in Akaike (1974).

The AIC is defined as follows:

$$\mathrm{AIC} = 2k - 2\ln(\mathcal{L}) \tag{12}$$

where $k = p + q + d$ refers to the number of parameters in the statistical model and $\mathcal{L}$ is the maximum likelihood of the model.

The statistical model with the lowest AIC is considered the best model Shumway et al. (2000).

3.3 GAN-based Methods for Anomaly Detection in Time Series

3.3.1 Motivation for selection of the State-of-the-Art GAN-based Methods for Anomaly Detection in Time Series

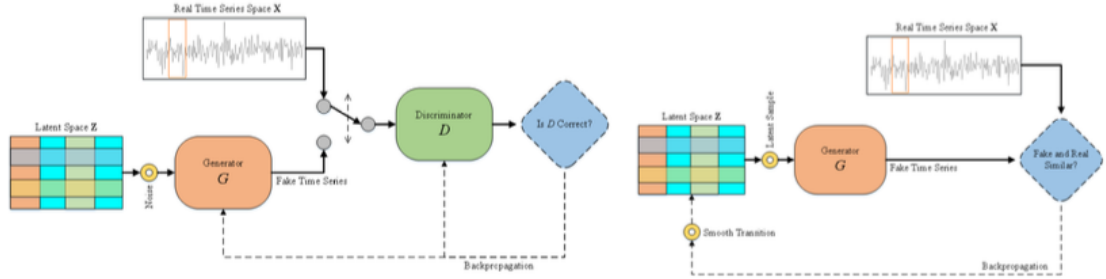
This study decided to focus on CAN-based methods for detecting anomalies in univariate time series. Due to these constraints, only a few algorithms using a GAN-based approach exist: TAnoGAN Bashar and Nayak (2020), TadGAN Geiger et al. (2020). MAD-GAN Li et al. (2019) and LSTM-VAE-GAN Niu et al. (2020), GAN-AD Li et al. (2018) were developed for anomaly detection in multivariate time series; therefore, they were not considered in the scope of this study. It was decided to use two methods: TAnoGAN, representing a classical GAN-based approach, and TadGAN, which presents a brand new architecture inspired by GAN principles. In the following Chapters 3.3.2 3.3.3, TAnoGAN and TadGAN are described in detail.

3.3.2 TAnoGAN: Time Series Anomaly Detection with Generative Adversarial Networks

Time Series Anomaly Detection with Generative Adversarial Network (TAnoGAN) Bashar and Nayak (2020) represents a method to detect anomalies in univariate time series on data sets with a small number of observations. It is based on a classical GAN architecture with the following key components, namely the Generator, the Discriminator and their interconnections. Figure 12 shows an architecture of TAnoGAN, where a) represents the process of training the generator and discriminator, and b) describes the process of mapping the real data into the latent space.

In the case of detecting anomalies in short time series sequences, Bashar and Nayak (2020) propose the following configuration for the generator and discriminator:

1. **Generator** $\mathcal{G}$ is LSTM with three layers containing 32, 64 and 128 hidden units respectively.



(a) Representing Normal Time Series with Generator G (b) Mapping Real-Data to the Latent Space

Figure 12: TAnoGAN Algorithm: a) The process of adversarial training. b) The process of anomaly detection after training. Source: Bashar and Nayak (2020)

.

2. **Discriminator** $\mathcal{D}$ is a single-layer LSTM with 100 hidden units.

The anomaly detection process using TAnoGAN includes the following steps:

Data preparation: The original study utilises the NAB dataset Ahmad et al. (2017), comprising 46 real time series signals selected for both training and evaluation purposes. Each signal was manually labelled with ground truth information regarding anomalies provided by Numenta. This documentation contained precise temporal information regarding anomalies' initiation and termination time points within each signal. For each signal from the datasets the data is normalised in order to process it to the range [-1, 1], then each signal was partitioned into smaller sub-sequences using a sliding window of size $s_w = 60$ and step $s_s = 1$.

Training the generator and discriminator: The training process is an adversarial training process that was described in detail in Chapter 2.4.

Anomaly detection: Anomaly detection requires mapping the real time series sequence $\mathbf{x}$ into the latent space $\mathbf{z}$ to estimate its similarity to the generated data. The mapping process involves randomising $\mathbf{z}^{(1)}$, $\mathcal{G}(\mathbf{z}^{(1)})$ and updating the parameters to find the optimal $\mathbf{z}^{(2)}$ and repeat for $i = \{1, ..., \Lambda\}$ steps. The loss function $\mathcal{L}$ for mapping $\mathbf{x}$ and $\mathbf{z}$ is defined as the weighted sum of the residual loss $\mathcal{L}_R$ and the discrimination loss $\mathcal{L}_D$.

The residual loss $\mathcal{L}_R$ is computed for the point-wise difference between the real and generated subsequence and is defined as

$$\mathcal{L}_R(\mathbf{z}^{(\lambda)}) = \sum_{\lambda=1}^{\Lambda} |\mathbf{x} - \mathcal{G}(\mathbf{z}^{(\lambda)}|) \tag{13}$$

The discrimination loss $\mathcal{L}_D$ is calculated as

$$\mathcal{L}_D(\mathbf{z}^{(\lambda)}) = \sum_{\lambda=1}^{\Lambda} |f(\mathbf{x}) - f(\mathcal{G}(\mathbf{z}^{(\lambda)})| \tag{14}$$

The total loss function $\mathcal{L}$ is calculated using the following equation:

$$\mathcal{L}(\mathbf{z}^{(\lambda)}) = (1-\gamma)\mathcal{L}_R(\mathbf{z}^{(\lambda)}) + \gamma\mathcal{L}_D(\mathbf{z}^{(\lambda)}) \tag{15}$$

The anomaly estimates $A(\mathbf{x})$ can be calculated from the loss function $\mathcal{L}$:

$$A(\mathbf{x}) = (1-\gamma)R(\mathbf{x}) + \gamma D(\mathbf{x}) \tag{16}$$

where the residual estimate $R(\mathbf{x}) = \mathcal{L}_R(\mathbf{z}^{(\Lambda)}$ and the discrimination estimate $D(\mathbf{x}) = \mathcal{L}_D(\mathbf{z}^{(\Lambda)}$. The parameter γ and threshold border is empirically chosen. A high anomaly score $A(\mathbf{x})$, suggests the presence of an anomalous short sequence. In contrast, a low anomaly score indicates a short sequence that closely aligns with the overall data distribution $\mathbb{X}$ learned by the generator $\mathcal{G}$ during adversarial training.

3.3.3 TadGAN: Time Series Anomaly Detection Using Generative Adversarial Networks

Time-series Anomaly Detection using Generative Adversarial Network (TadGAN) Geiger et al. (2020) represents a potentially new algorithm developed to detect anomalies in time-series signals. This algorithm provides a closed-loop system consisting of several components that work together to detect anomalies in time series data. The TadGAN architecture consists of the following components:

1. **Generator** $\mathcal{E}$ is a single-layer bidirectional LSTM with 100 hidden units. It converts input time series $\mathbf{x}$ into latent representation vectors $\mathbf{z}$ in latent space $\mathbb{Z}$.

2. **Generator** $\mathcal{G}$ is a two-layer bidirectional LSTM network with 64 hidden units in each layer. It acts as a decoder and generates time series based on the latent representation of $\mathbf{z}$.

3. **Discriminators** C_x and C_z are two one-dimensional convolutional networks. They perform row reconstruction quality assessment (C_x evaluates how well the signal $\mathbf{x}$ is reconstructed by the generator $\mathcal{G}$) and hidden representation similarity assessment (C_z evaluates how well the hidden representation $\mathbf{z}$ matches the original signal $\mathbf{x}$).

Figure 13 shows the main components and relationships of TadGAN.

Based on TadGAN architecture, two generators $\mathcal{E}$, and $\mathcal{G}$ are responsible for representing and generating the time series, respectively, and two discriminators C_x and C_z evaluating the quality of the reconstruction and the correspondence of the representation can be distinguished.

TadGAN endeavours to train itself to generate realistic time series in latent space while simultaneously evaluating the quality of the reconstruction and the correspondence between the original series and their representations. This allows the model

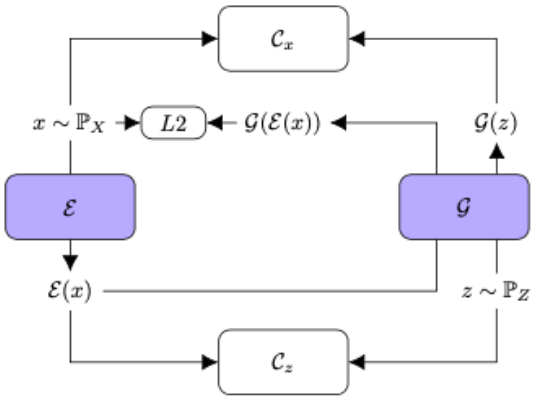


Figure 13: TadGAN Architecture: $\mathcal{E}$, refer to Encoder, $\mathcal{G}$ refer to Decoder, C_x, C_z are critics. Source: Geiger et al. (2020)

to detect anomalies in time series by comparing the reconstructed series with the original data and evaluating the differences between latent representations.

The core principle of TadGAN relies on analysing two mapping functions that connect two domains, $\mathbb{X}$ and $\mathbb{Z}$: one is $\mathcal{E} : \mathbb{X} \to \mathbb{Z}$, and the other is $\mathcal{G} : \mathbb{Z} \to \mathbb{X}$. The domain $\mathbb{X}$ represents the real input data. The domain $\mathbb{Z}$ represents the latent space, where random vectors $\mathbf{z}$ are chosen to represent white noise, $\mathbf{z} \sim \mathcal{N}(0, 1)$.

Using these mapping functions, it becomes feasible to reconstruct a signal with: $x^{(i)} \to \mathcal{E}(x^{(i)}) \to \mathcal{G}(\mathcal{E}(x^{(i)})) \sim \hat{x}^{(i)}$. Adversarial learning techniques were utilized to train the mapping functions $\mathcal{E}$ and $\mathcal{G}$. The function $\mathcal{E}$ serves a dual purpose as an encoder, transforming time series sequences into a latent space. In contrast, $\mathcal{G}$ operates as a decoder, converting the latent space representation into a reconstructed time series.

Throughout the training process, the primary aim of the adversarial critic C_x is to distinguish between real signals from domain $\mathbb{X}$ and the generated time series data created by $\mathcal{G}(\mathbf{z})$. Concurrently, another adversarial critic, C_z, evaluates the quality of the mapping into the latent space $\mathbb{Z}$.

The anomaly detection process includes the following steps:

Data preparation: The data is first normalized for each signal within the datasets, bringing it into the range of [-1, 1]. Subsequently, an appropriate interval is selected to aggregate the data. For this purpose, the window size $s_w = 100$ and the step $s_t = 1$ were chosen.

The training process consists of the following steps:

1. For each value of k from 0 to n_{critic} (where n_{critic} is the number of training iterations for the critic)

2. Select m real time series $(x_i^{1 \cdots t})_{i=1}^m$ from the training sample.

3. Generate m random vectors $(z_i^{1\dots k})_{i=1}^m$ from the latent space.

4. Compute gradients for the critic C_x on its parameters w_{C_x} in order to reduce the loss function. This loss function includes the Wasserstein-1 distance and the gradient penalty. Then, update the parameters of the critic w_{C_x} using the Adam optimiser.

$$g_{w_{C_x}} = \nabla_{w_{C_x}}[\frac{1}{m}\sum_{i=1}^{m} C_x(x_i) - \frac{1}{m}\sum_{i=1}^{m} C_x(\mathcal{G}(z_i)) + gp(x_i, \mathcal{G}(z_i))] \tag{17}$$

$$w_{C_x} = w_{C_x} + \nu \cdot adam(w_{C_x}, g_{w_{C_x}}) \tag{18}$$

5. Compute gradients for the critic C_z on its parameters w_{C_z} to reduce the loss function. This loss function includes the Wasserstein-1 distance and the gradient penalty. Then, update the parameters of the critic w_{C_z} using the Adam optimiser.

6. Then resample m real time series $(x_i^{1\dots t})_{i=1}^m$ from the training sample.

7. Generate m random vectors $(z_i^{1\dots k})_{i=1}^m$ from the latent space.

8. Compute the gradients for the generators $\mathcal{G}$ and $\mathcal{E}$ with respect to their parameters $w_{\mathcal{G},\mathcal{E}}$, seeking to minimise a loss function that includes Wasserstein-1 distance and loop consistency loss.

9. Update the parameters of the generators $w_{\mathcal{G},\mathcal{E}}$ using the Adam optimiser.

Anomaly detection: For anomaly detection Geiger et al. (2020) propose three ways to calculate the reconstruction error (RE) between real signal $\mathbf{X}^{(i)}$ and reconstructed signal $\hat{\mathbf{X}}^{(i)}$, such as a point-wise difference, an area difference, and dynamic time warping (DTW):

- Point-wise difference (point) computing the discrepancy between the actual value and the predicted value for each time series observation,

$$s_t = |x^{(t)} - \hat{x}^{(t)}| \tag{19}$$

- Area difference (area) is defined as the average difference between areas:

$$s_t = \frac{1}{2 * l}|\int_{t-l}^{t+l} x^{(t)} - \hat{x}^{(t)}\mathrm{d}x| \tag{20}$$

  This estimation method effectively identifies regions characterised by minor variations that remain constant over extended temporal intervals.

- Dynamic Time Warping (DTW) aims to find the optimal alignment between two time series. It is performed by finding the shortest path in the distance matrix between pairs of elements. This path is chosen to minimise the total distance between the corresponding elements of the rows.

$$s_t = W^* = DTW(\mathbf{X}, \hat{\mathbf{X}}) = \min_W [\frac{1}{K} \sqrt{\sum_{k=1}^{K} w_k}] \tag{21}$$

  W^* represents the optimal alignment between real and reconstructed time series elements.

To compute Anomaly score Geiger et al. (2020) proposed to combine a C_x error and RE. The first step involves computing the z-scores for both errors, denoted as $Z_{C_x}(x)$ and $Z_{RE}(x)$, using the Z-Transformation formula Bronstein et al. (2012). Secondly, one of the methods for combining both scores was employed:

- **Convex combination**:

$$A(x) = \alpha Z_{RE}(x) + (1 - \alpha) Z_{C_x}(x) \tag{22}$$

  α shows the importance of both errors and is chosen empirically

- **Multiplication**:

$$A(x) = \alpha Z_{RE}(x) \times Z_{C_x}(x) \tag{23}$$

The research conducted in Geiger et al. (2020) demonstrated that the most effective approach for anomaly detection involved a convex combination of the C_x critic error and DTW. This particular combination will be utilised to assess the performance of TadGAN in a replication study. The results of this evaluation are presented in Chapter 5.

3.4 Evaluation Metrics

To evaluate the quality of the chosen methods the standard evaluation metrics used in statistics and machine learning will be employed: accuracy, precision, recall and F_1-Score.

To compute the primary metrics, a confusion matrix is required, a table where the real values are represented horizontally, and the values predicted by the model are represented vertically. In the context of anomaly detection, the true values are taken as information regarding the presence of an anomaly (P) or the absence of an anomaly (N). The predicted values of model are classified as positive predictions (PP) if the model identified a measurement as an anomaly and as negative predictions (NP) if absence of anomaly was detected.

The confusion matrix is presented in Table 1.

Table 1: Confusion matrix

	P	N
PP	True Positive (TP)	False Positive (FP)
NP	False Negative (FP)	True Negative (TN)

More specifically, TP represents cases where the model correctly classified an anomaly as an anomaly. TN represents cases where the model correctly did not detect an anomaly. This means that the model correctly classified the absence of an anomaly as the absence of an anomaly. FP represents cases where the model incorrectly classified data as anomalies when there is no real anomaly. FN represents cases where the model incorrectly classified normal data as anomalies, meaning the model missed an anomaly and predicted that there were no anomalies when they actually existed.

Based on the confusion matrix, fundamental performance metrics for the chosen anomaly detection methods can be calculated.

Accuracy is used as a statistical measure of how well the model correctly identifies a particular state. It is computed as the ratio of correctly classified observations to the total number of classified observations Goutte and Gaussier (2005). In scientific terms, accuracy assesses the model's overall performance in correctly identifying states and is calculated as follows

$$Accuracy = \frac{TP + FP}{TP + FP + FN + TN} \tag{24}$$

Precision quantifies the model's ability to make precise positive predictions Goutte and Gaussier (2005). In scientific terms, precision is the ratio of the number of true positive predictions (True Positives, TP) to the total number of positive classes predicted (True Positives + False Positives, $TP + FP$).

$$Precision = \frac{TP}{TP + FP} \tag{25}$$

Recall measures a model's ability to correctly classify all positive examples out of the total number of actual positive examples. It quantifies how well the model can capture all positive cases in a classification task Goutte and Gaussier (2005). Recall is defined as the ratio of the number of true positive predictions (TP) to the total number of objects actually belonging to a given class (True Positives + False Negatives, $TP + FN$). In scientific terms, it is expressed as follows:

$$Recall = \frac{TP}{TP + FN} \tag{26}$$

A higher Recall value indeed indicates that the model is better at detecting positive cases, which is particularly important in situations where missing positive values could have significant consequences, such as in signal anomaly detection tasks.

Recall and precision exhibit an inverse relationship: increasing Recall can lead to a decrease in Precision, and vice versa. When striving to enhance Precision, the model tends to classify only the most obvious examples as positive. This selective approach may lead to a reduction in recall but an improvement in accuracy for the instances classified as positive. Conversely, when recall is prioritised and increased, the model may classify a greater number of observations as positive, resulting in an increased TP rate but a decreased overall recall.

The F_1-Score combines precision and recall using a harmonic mean, which is particularly useful when precision and recall have different values. The F_1-Score balances the importance of both precision and recall

$$F_1\text{-}Score = 2 * \frac{Precision * Recall}{Precision + Recall}$$

A high F_1-Score value indicates a good balance between precision and recall, which is important when both the false-positive and false-negative predictions are significant.

4 Datasets

This chapter pertains to the analysis of selected datasets, which are used to answer the research questions in this study. Chapter 4.1 provides motivation for choosing the selected datasets for the study. Chapter 4.2 provides general characteristics and metrics for the investigation of the chosen NAB and WD datasets. Chapter 4.3 describes the general characteristics of the labelled Numenta Anomaly Benchmark (NAB) dataset, while Chapter 4.4 outlines the Weather Dataset's (WD) preparation process and general characteristics.

4.1 Motivation

As mentioned earlier in Chapter 2.2.2, a limited number of datasets are available for evaluating the performance of new anomaly detection methods for time series data. For checking a performanz of new anomaly detection algorithms in univariate time series researchers Geiger et al. (2020), Bashar and Nayak (2020) used Numenta Anomaly Benchmark (NAB) from Numenta Company Ahmad et al. (2017) and S5 - A Labelled Anomaly Detection Dataset (Yahoo) from Yahoo! Webscope. [3] The Yahoo dataset is provided by Yahoo Company for non-commercial research purposes on demand. It consists of four different collections of signals, three of which are synthetic data. The majority of anomalies in this dataset are point anomalies.

The NAB dataset primarily comprises real-world data obtained from various sources. Signals in the NAB collection have varying lengths, ranging from 1,127 to 22,695 points. In their study Bashar and Nayak (2020) postulated that TAnoGAN works better for smaller sequences than for larger ones. Therefore, choosing the NAB dataset can help to test this hypothesis.

Additionally, for this study, an unlabeled dataset named the Weather Dataset (WD) was created. It comprises temperature and pressure data collected in Bamberg from 1961 to 2020, sourced from the *Open data area of Climate Data Centre (CDC) of the Deutscher Wetterdienst (DWD).*[4]

The NAB dataset, primarily composed of real-world data with varying characteristics, was utilised to replicate the results of the selected state-of-the-art GAN-based methods, TAnoGAN and TadGAN. Furthermore, the NAB dataset served as a means to assess the performance of these methods in relation to the types of stationarity present in their signals. The WD dataset was utilised to detect potential weather anomalies using the classical statistical ARIMA and the state-of-the-art GAN-based TadGAN methods.

[3] https://shorturl.at/muOX2

[4] https://www.dwd.de/DE/leistungen/cdc/cdc_ueberblick-klimadaten.html

4.2 General Characteristics

The NAB and WD datasets have several univariate time series (referred to as signals), which are grouped based on certain general features within the collections. The WD dataset contains two collections (temperature and pressure) of unlabelled signals, the NAB dataset contains 6 collections of signals with labelled anomalies.

For each signal within each collection across the chosen datasets, the following parameters were investigated: the number of signals, the total number of data points, and two boolean parameters indicating whether the signals are real or artificial and whether the dataset contains anomalies or loss signals.

Additional parameters were counted for the NAB dataset, which included anomaly annotations. These parameters encompassed determining the total amount of anomalies, the amount of anomaly points per collection, calculating the average amount of anomaly points within each group within the collection, and establishing the percentage of anomaly points in the collection. The detailed information for each investigated parameter is provided in Table 3 for the NAB dataset and Table 6 for the WD dataset.

Signal Each signal is represented as an individual file with the extension .csv, containing two key data fields: *timestamp*, denoting the time of registered measurement, and *value*, representing the numerical value of the signal at the timestamp.

Amount of signals The number of signals in each collection was counted empirically and defined as the number of signals presented as .csv files in each collection folder. For this purpose, a scan of the file structure of each dataset was conducted, and the total number of files with the .csv extension in each folder was counted.

Amount of data points To determine the amount of data points in each collection, an analytical assessment of the length of each signal in that collection was performed. The length of a signal is defined as the number of time points contained within it. Subsequently, the total sum of the lengths of all signals within the collection was calculated as follows:

$$\text{amount of data points} = \sum_{i=1}^{N} length(\mathbf{X}^i) \tag{27}$$

where $\mathbf{X}^i$ represents the i-th signal, and N is the amount of signals within each collection.

Amount of anomaly points The total number of anomalous measurements was calculated by summing all measurements across each signal within a collection, where the anomaly label was set to 1, indicating an anomalous value at that time point.

$$\text{amount of anomaly points} = \sum_{i=1}^{N} length(\mathbf{X}^i \cdot \mathbb{I}_{\text{is\_anomaly}}) \tag{28}$$

Where $\mathbf{X}^i$ represents the i-th signal, N is the total number of signals within a specific collection and $\mathbb{I}_{\text{is\_anomaly}}$ is an indicator function defined as:

$$\mathbb{I}_{\text{is\_anomaly}}(X) = \begin{cases} 1, & \text{if is\_anomaly} = 1 \\ 0, & \text{is\_anomaly} = 0. \end{cases} \tag{29}$$

Percentage of anomaly points The percentage of anomalous data points was determined by dividing the number of anomalous data points by the total number of data points in the dataset, and then expressing this ratio as a percentage.

$$\%\text{of anomalies} = \frac{\text{amount of anomaly points}}{\text{amounts of data points}} * 100\% \tag{30}$$

Amount of anomalies and its types The number of anomalies was determined as follows: Each signal $\mathbf{X}^{(i)}$ within a collection was inspected to identify measurements denoted as $\mathbf{X}_t^{(i)}$ with an anomaly flag of 1. Subsequently, the next measurement, $\mathbf{X}_{t+1}^{(i)}$, was assessed. If its anomaly flag was 0, $\mathbf{X}_t^{(i)}$ was classified as a point anomaly. If the anomaly flag was also 1, the search continued until a point was reached where measurement $\mathbf{X}_{t+j}^{(i)}$ met the following conditions: all anomaly labels for measurements from $\mathbf{X}_t^{(i)}$ to $\mathbf{X}_{t+j-1}^{(i)}$ were 1, and the label for $\mathbf{X}_{t+j}^{(i)}$ was 0. The set of measurements $\{\mathbf{X}_t^{(i)}, \ldots, \mathbf{X}_{t+j-1}^{(i)}\}$ constituted a group anomaly, with its length representing the extent of the group anomaly. Similar analyses were conducted for each signal within a collection, and the total count of each anomaly type within the collection was obtained by summing the point and group anomalies for each signal.

$$\text{amount of anomalies} = \text{amount of point anomalies} + \text{amount of group anomalies} \tag{31}$$

Amount of anomaly points pro group To calculate this parameter, firstly the number of anomalous points covered by group anomalies is determined. This is achieved by subtracting the amount of point anomalies from the overall amount of anomalous data points. Subsequently, the residual number of anomalous data points is divided by the previously established amount of group anomalies.

$$\text{amount of anomaly point/group} = \frac{\text{amount of data point–amount of point anomalies}}{\text{amount of group anomalies}} \tag{32}$$

Stationarity Each signal underwent a stationarity analysis using the ADF Dickey and Fuller (1979) and KPSS Kwiatkowski et al. (1992) tests, as detailed in Chapter 3.1. Two approaches were employed in this study. The first one utilised a combination of ADF and KPSS tests with various regression parameters available for both tests, while the second one relied on the results of the ADF test only.

In the case of the combination of ADF and KPSS tests, the regression parameters refer to the coefficients of the regression equation estimated during the ADF test, which can include a constant (c) or a constant and a linear trend (ct). The results are also stored in separate tables for each of the selected datasets: NAB in Table 4. In this approach to the stationarity of signals, the signals were divided into three categories: TS-Stationary, Stationary, and DS-Stationary.

- TS-Stationary, also known as Trend Stationary: In this category, signals are considered stationarity after removing the linear trend component but may still contain seasonal components Hamilton (1994).
- Stationary: This category includes signals with a DGP that is not easily categorised as either TS-Stationary or DS-Stationary. These signals may generally contain non-linear (exponential or quadratic) trends or may result from a DGP that does not include a trend component. In this study, these time series are considered stationary because the ADF test outperforms the KPSS test. For this reason, signals are considered stationary without further specification.
- DS-Stationary signals: These are signals that are not stationary initially but can be transformed into stationary ones through a differencing procedure Hamilton (1994).

The mathematical definitions of TS- and DS-Stationarity concepts were presented earlier in Chapter 2.1.3.

To classify each signal into one of the three types, the following rules, as shown in Table 2, were employed.

Table 2: An approach for harmonising the outcomes of time series testing using the ADF and KPSS tests.

ADF \ KPSS	H_0: Trend stationary	H_1: Unit-root
H_0: Unit-root	low test power; recommends considering alternative statistical tests or visual inspection	DS-Stationary signal
H_1: No unit-root	TS-Stationary signal	Stationary signal

The second approach for determining stationarity involved using the ADF test independently. This test can be conducted using four different regression parameter

options: no constant, no trend (n); constant (c); constant and trend (ct); and constant and quadratic trend (ctt). By using only this test, it is possible to determine whether a signal is stationary (indicating the absence of a unit root) or non-stationary (suggesting the presence of a unit root). The results of this test for the two investigated datasets are presented in Table 5 for the NAB dataset and in Table 9 for the WD dataset.

For providing ADF and KPSS tests, the maximum number of lags for computing a linear regression model was not limited. To determine whether the null hypothesis is rejected in the both ADF and KPSS tests in this study, a significance level of p-value $= 0.05$ was utilised. If p-value < 0.05, it is possible to reject the null hypothesis H_0 and conclude that the alternative hypothesis H_1 is true Chan (2004). Conversely, if p-value $>= 0.05$, it is not possible to reject the null hypothesis. If the p-value approaches 0.05, the calculated ADF and KPSS statistics are compared with the ADF and KPSS critical values at the 5% significance level. H_0 can be rejected if $ADF_{\text{value}} < ADF_{5\%}$ Dickey and Fuller (1979) and $KPSS_{\text{value}} > KPSS_{5\%}$ Kwiatkowski et al. (1992)

The following Chapters 4.3, 4.4 will describe NAB and WD datasets in detail.

4.3 Numenta Anomaly Benchmark (NAB) Dataset

Numenta Anomaly Benchmark (NAB) [5] is a dataset created by Numenta Ahmad et al. (2017). This dataset was explicitly developed to evaluate the performance of anomaly detection algorithms in time series data. It contains diverse univariate time series that include various anomalies, such as lost signals, equipment failures or artificially simulated anomalies. The NAB dataset consists of 58 time series signals, which have been categorised into seven collections. This thesis utilises 53 datasets from six collections, containing anomalies. The *ArtificialWithoutAnomalies* collection, which contains the remaining six signals without anomalies, was not considered in this work. The ArtificialWithAnomaly (Art) collection contains artificially generated time series with predefined anomalies. The RealAWSCloudwatch (AWS) collection contains Amazon Web Services server metrics such as CPU utilisation and network activity volumes measured in different periods, for different signals, between October 2013 and April 2014. The RealAdExchange (AdEx) collection includes click metrics for online advertising, such as *cost per click* (CPC) and *cost per thousand impressions* (CPM) measured from July to September 2011. The RealKnownCause (Known) collection contains temporal signals from different sources with known anomalies in advance. These signals were measured at different time periods between March 2013 and January 2015. The RealTraffic (Traffic) collection contains traffic congestion, speed, and travel time data from the Minnesota Department of Transportation in the Twin Cities region measured for different intervals between July and September 2015. The RealTweets (Tweets) collection includes

[5] https://github.com/numenta/NAB/tree/master/data

the number of mentions of public companies in tweets over 5-minute intervals from February to April 2015. [6]

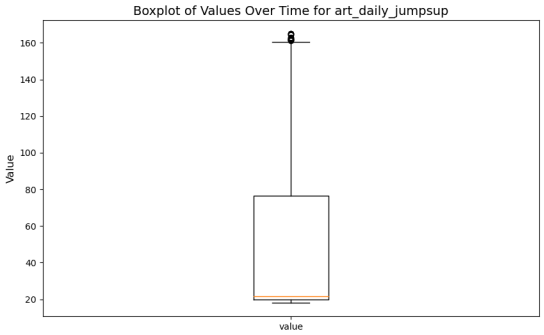
The length of each time signal ranges from 1,127 to 22,695 data points. The total number of data points in the NAB dataset is 3,655,551 points Ahmad et al. (2017), with 9% of the data points representing abnormalities. Each time signal is represented as a .csv file and contains measured values (*value*) recorded at specific points of time (*timestamp*). Anomalies are provided in separate *combined_windows.JSON* files, which include timestamps indicating the start and end of abnormal behaviour.[7] As part of this research work, each time signal has been extended with an *is_anomaly* measurement, which provides a binary classifier that assigns 1, if the time period is specified as anomalous in the *combined_windows.JSON* file and 0 otherwise.

To initially explore the data, particularly when working with real-world data collected from different sources like in the NAB dataset, boxplot DuToit et al. (2012) refers to valuable tools. The main elements of a boxplot are as follows: The horizontal orange line on the boxplot (see Figure 14b) represents the median, the upper limit of the black box indicates the third quartile (Q3), and the lower limit shows Q1. The upper horizontal line (whisker) points to the maximum value that is not considered an outlier (typically within 1.5 times the interquartile range (IQR) from Q3). The lower whisker points to the minimum value that is also not considered an outlier (typically within 1.5 times the IQR from Q1), where IQR refers to the interquartile range, representing the difference between the boundaries of Q3 and Q1. Points that fall outside the whiskers are considered potential anomalies (outliers).As observed in Figure 14, boxplots display significant differences within the choosing signals from difference collection of the NAB dataset, indicating variations in data distributions. This implies that there are signals with different properties within the dataset. Boxplots for all signals in six collections of the NAB dataset are provided in the Appendix A.

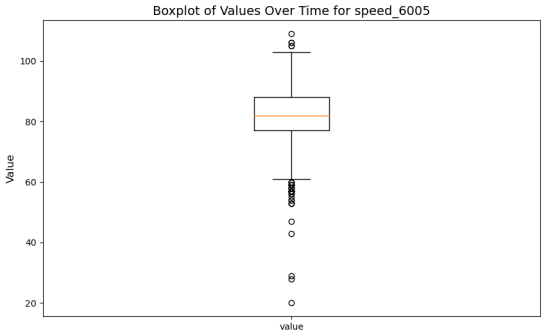
The range of data based on the information obtained from boxplots (see Figure 14 and Figures in A) also varies. The datasets in the Art (see Figure 14a) and Trafic (see Figure 14b) collections have the widest range of data falling within the normal distribution (between the $25th$ percentile, Q1, and the $75th$ percentile, Q3)). In contrast, the data from the AWS (see Figure 14c) and Tweets (see Figure 14d) collections have a minimal range of data distributed between Q1 and Q3 but exhibit a substantial dispersion between the minimum and maximum data values. This may indicate a large number of potential anomalies in signals. The table 3 provides evidence supporting this hypothesis. It is observed that the *AWS* and *Tweets* collections contain the highest number of anomalies within the NAB dataset: 26 and 33 anomalies, respectively.

The NAB summary, which includes overview parameters described in Section 4.2, is provided in Table 3.

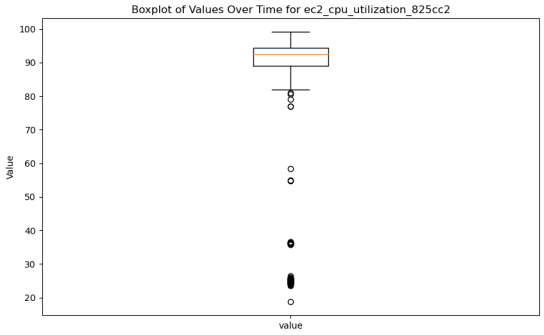
[6] https://github.com/numenta/NAB/blob/master/data/README.md
[7] https://github.com/numenta/NAB/blob/master/labels/combined_windows.json



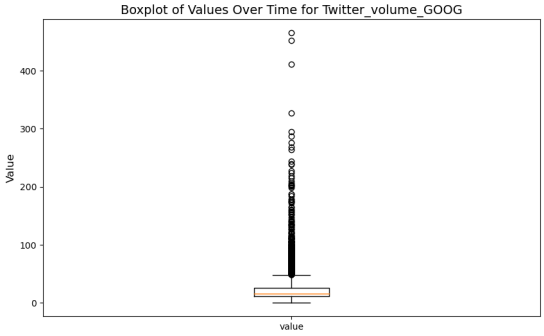
(a) Boxplot of an *art_daily_nojump* signal from *Art* collection



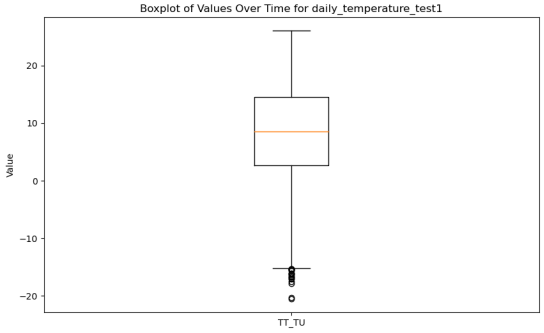
(b) Boxplot of a real *speed_6005 signal* from $Traffic$ collection



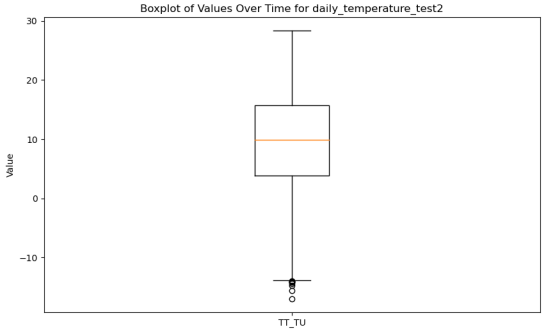
(c) Boxplot of a real signal from AWS collection



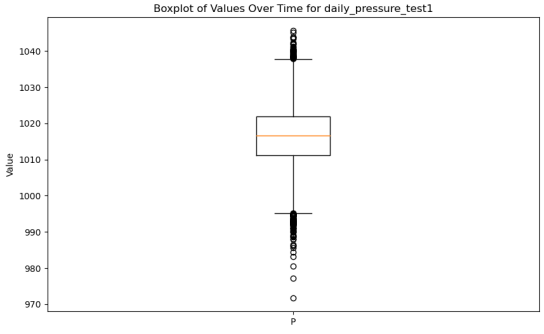
(d) Boxplot of a real *Twitter_GOOG* signal from $Tweets$ collection



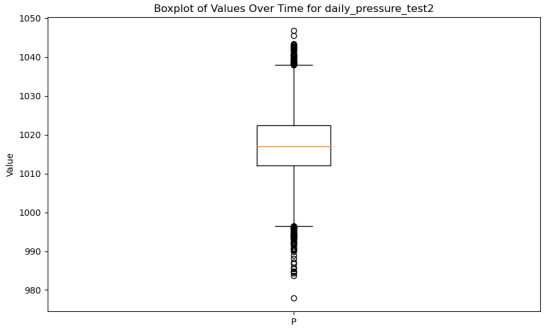
(e) Test 1 Temperature 01.01.1961 - 31.12.1980



(f) Test 2 Temperature 01.01.2001 - 31.12.2020



(g) Test 1 Pressure 01.01.1961 - 31.12.1980



(h) Test 2 Pressure 01.01.2001 - 31.12.2020

Figure 14: Boxplots of selected signals from the NAB dataset (a) -(d). Boxplots for daily average Temperature and Pressure from WD dataset (e) - (h). The horizontal orange line represents the median, the upper limit of the black box indicates Q3, and the lower limit shows Q1. The upper and lower whiskers point to borders for normal data. Points refer to potential anomalies.

Table 3: NAB dataset summary

Property	Art	AdEx	AWS	Known	Traf	Tweets
amount of signals	6	6	17	7	7	10
amount of data points	24,191	9,610	67,740	69,561	15,664	158,631
amount of anomalies	6	14	26	19	14	33
point	0	0	0	0	0	0
group	6	14	26	19	14	33
amount of anomaly points/set	1,449	909	6,186	6,594	1,560	15,651
average amount of anomaly points/group	242	65	238	347	111	474
% of anomaly points	5.99%	9.46%	9.13%	9.48%	9.96%	9.87%
is synthetic?	yes	no	no	no	no	no
lost signals?	no	no	yes	no	no	no

There is one collection (Art) that contains synthetically generated signals with different types of anomalies, while the other five collections include signals that include real data obtained from various sources. There is one collection (AWS) that contains anomalies of the *lost signal* type, signifying periods when signal values were not recorded.

As shown in Table 3, the AWS collection contains the highest number of signals (17), while the AdEx collection has the lowest number (6). However, regarding the overall count of data points, the Tweets collection takes the lead, housing signals with the highest number of data points (158,613). On the other hand, the AdEx collection contains the least amount of measurements within the signals (9,610).

It is also shown that all anomalies in the NAB dataset are categorised as group anomalies, meaning they have a length greater than one. Additionally, the percentage of anomalous data points relative to the total number of points in the collection is slightly lower for synthetic data in the Art collection, amounting to 5.99%. In other collections that contain real data, this percentage is slightly higher, averaging around 9.5%. The Traf collection stands out with the highest percentage of anomaly points within all data points in the collection, at 9.96%.

Although both the AdEx and Traf collections have the same number of group anomalies (14), the average number of anomaly points in each group is greater for Traf (111) than for AdEx (65). This correlation can be attributed to the difference in the amount of data points within the collections: Traf has more data points (15,664) than AdEx (9,610). Table 3 shows that the average amount of anomaly points in a group within a collection is directly proportional to the total number of data

points in the collection, and it is not correlated with the number of group anomalies. This observation can significantly impact the performance of window-based anomaly detection methods.

Next, the question of stationarity will be discussed. As mentioned earlier, tests for stationarity in time series are not governed by an absolute truth, and the outcomes can depend on the parameters of the chosen test. The same time series signal can be classified as both TS-Stationary and Stationary depending on the test parameters. Furthermore, by limiting the maximal number of lags in statistical tests, the signal may also be classified as DS-Stationary based on the corresponding test results.

Table 4 illustrates that, for some collections, the choice of regression parameter is decisive when classifying data as TS-Stationary or Stationary.

Table 4: Amount of TS-Stationary (TS-Stat.), DS-Stationary (or non-stationary) (DS-Stat.) and Stationary signals with a non-linear trend (Stat.) based on the combination of ADF and KPSS tests under constant and constant with trend regression model in NAB Dataset

	Constant			Constant and trend		
	TS-Stat.	Stat.	DS-Stat.	TS-Stat.	Stat.	DS-Stat.
Art	5	1	0	5	1	0
AdEx	1	3	2	2	3	1
AWS	9	3	5	4	8	5
Known	2	5	0	0	7	0
Traf	3	4	0	2	5	0
Tweet	7	3	0	7	3	0

Based on Table 4, it is evident that artificial signals from the Art collection and signals from real data in the Tweets collection are insensitive to changes in the criterion. For the AWS collection, the signal *exchange-2_cpc_results* shifts from being DS-Stationary to TS-Stationary after changing the regression parameter in tests from *Constant* to *Constant and trend*. In the remaining collections, changing the regression parameter from *Constant* to *Constant and trend* results in a decrease in the number of TS-Stationary signals but an increase in the number of Stationary signals generated by DGP with nonlinear trends or other types of trends.

Table 4 illustrates that the assumption of whether a signal is stationary depends on the parameter settings of both ADF and KPSS tests. Therefore, in cases of uncertainty, it is necessary to employ alternative tests or graphical methods.

In this study, an alternative method for assessing the stationarity of signals involves exclusively using the ADF test. As previously mentioned, no constraints were placed on the maximum lag, and all available regression parameters (c, ct, ctt, and n) were

considered. Under this approach, signals were categorised as either Stationary or Non-stationary.

According to Table 5, the largest number of non-stationary signals was identified when using the regression parameter n. A total of 21 signals were classified as Non-stationary according to the ADF test using this criterion. However, a completely different picture emerges when using the parameter c. In this case, only 7 signals from the dataset are Non-stationary. Interestingly, changing the regression parameter from *ct* to *ctt* had no influence on the number of Stationary and Non-stationary signals within all collections of the NAB dataset; the number of Non-stationary signals remained the same at 6. All these Non-stationary signals belong to the AWS and AdEx collection.

Table 5: Amount of Stationary (signals without a unit-root) and Non-Stationary (signals that contain a unit-root) based on ADF test under constant (c), constant with trend (ct), constant with linear and quadratic trend (ctt), without constant and trend (n) regression model in NAB Dataset

Property	Art	AdEx	AWS	Known	Traf	Tweets
Stationary (c)	6	4	12	7	7	10
Non-Stationary (c)	-	2	5	-	-	-
Stationary (ct)	6	5	12	7	7	10
Non-Stationary (ct)	-	1	5	-	-	-
Stationary (ctt)	6	5	12	7	7	10
Non-Stationary (ctt)	-	1	5	-	-	-
Stationary (n)	6	2	8	2	4	10
Non-Stationary (n)	-	4	9	5	3	-

Based on the data in Table 5, it can be observed that the Art and Twitts collections contain only stationary signals, and these signals are classified as stationary regardless of the type of regression used in the ADF test. In the other collections, conducting the ADF test with the parameter n classifies some stationary time series as non-stationary.[8]

4.4 Weather Dataset

The second dataset to be used in this study is the Weather Dataset (WD). This dataset is unlabelled, and the goal of this thesis with respect to the WD dataset is

[8]Detailed information about each signal and each parameter presented in the GitHub repository `https://github.com/anjahr/masterthesis-anomalie`

to explore the existence of anomalies utilising ARIMA (described in Chapter 3.2) and TadGAN algorithms (described in Chapter 3.3.3).

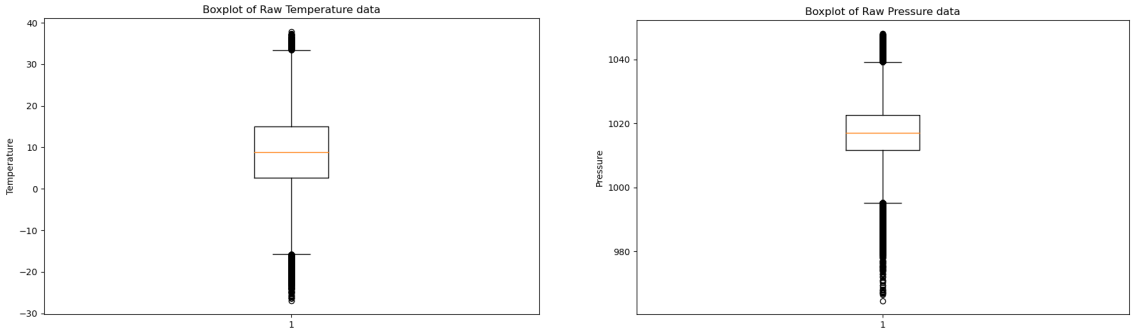
The data in the WD were extracted from the *Open data area of Climate Data Centre* (CDC) of the *Deutscher Wetterdienst* (DWD). [9] This research used data on weather temperature (temperature) and atmospheric pressure (pressure) changes in Bamberg, Germany, within the observation period 1960-2020. Namely, the data obtained from observation *station 282* located at *Dr Reimus-Sternwarte, Sternwartstraße 7, 96049 Bamberg*.

The time intervals for temperature measurements extended from 1961 to 2022, and for atmospheric pressure from 1949 to 2022. Temperature data measurements were performed hourly throughout the entire day for each day. Atmospheric pressure data measurements were taken every three hours during the day. Both of temperature and pressure raw data include details such as *STATIONS_ID*, which refers to a unique station identifier in the DWD catalogue (282) and *MESS_DATUM*, representing the date and time of measurement. Additionally, the temperature data contains *TT_TU* measurements, which signifies the air temperature in degrees Celsius, representing values measured at a height of 2 meters above the ground surface, and *RF_TU*, indicating the relative humidity in percentage. The pressure data contains *P0*, which represents the atmospheric pressure at the level of Station 282, and *P*, which represents the atmospheric pressure at sea level, which is used for this study. Temperature and pressure raw data also contain missing values, which are indicated in CDC raw data by a value of *-999*. Out of the $525,813$ measures from temperature raw data, only 130 were marked as missing, which is less than 0.02% of all data points. The linear interpolation method Bronstein et al. (2012) estimates missing values in a dataset by creating a straight-line connection between two known data points on either side of the missing value. Out of $344,768$ measurements from pressure raw data, $8,879$ were marked as missing, which is 2.58% of all data points. These missing measurements were imputed using the similar mechanism described for temperature.

As observed in Figure 15, both the temperature signal 15a and the pressure signal 15b exhibit a certain number of outliers, which can be considered as potential anomalies in the data. Notably, the normal range of values for the temperature signal (the distance between the whiskers on boxplot) is wider than the corresponding range for the pressure signal. Based on this, it can be inferred that the range of values considered normal for the temperature signal is broader than that for the pressure signal, which may indicate a higher prevalence of anomalies in the pressure signal.

The original data [10] contained temperature and data measurements spanning different observation periods. This thesis considered only the time frame from 01.01.1961 to 31.12.2020. Additionally, for further analysis and application of ARIMA Box

[9] https://opendata.dwd.de/climate_environment/)
[10] https://opendata.dwd.de/climate_environment/



(a) Boxplot of temperature raw signal (b) Boxplot of pressure raw signal

Figure 15: Boxplots for raw temperature and pressure signals from the WD dataset. The horizontal orange line represents the median, the upper limit of the black box indicates Q3, and the lower limit shows Q1. The upper and lower whiskers point to borders for normal data. Points refer to potential anomalies.

et al. (2015) and TadGAN Geiger et al. (2020) methods, preliminary preprocessing was conducted:

1. Data for the pressure signal were filtered based on the range defined for this study. Subsequently, the date format was converted into a calendar format for ease of use. To facilitate further work with the signal, columns containing unnecessary information for subsequent analysis, such as $STATIONS\_ID$, QN_8, RF_TU, and *eor* for the temperature signal, as well as $STATIONS\_ID$, QN_9, $P0$, and *eor* for the pressure signal, were removed.

2. Missing values (marked as -999 in the raw data) were imputed using an interpolation method.

3. The temperature and pressure signal values were aggregated on a daily basis, and the average daily temperature and pressure values were calculated.

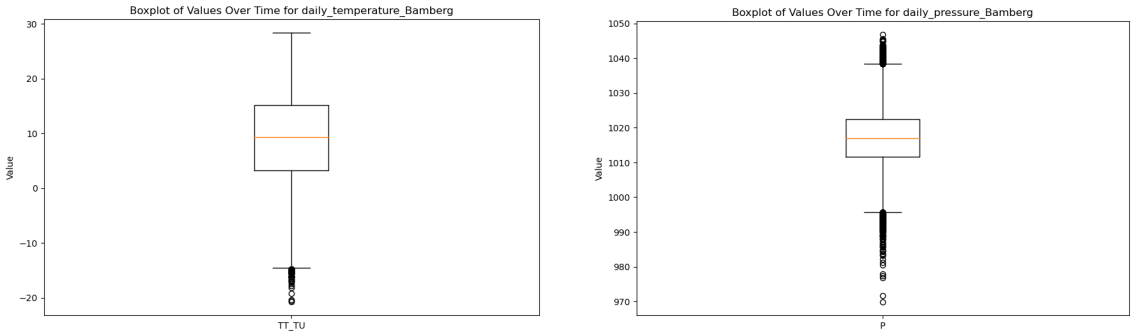
The pressure and temperature signals obtained at this stage contained information about the average daily temperatures and average daily pressure, respectively. To reduce the data volume, the temperature and pressure signals were then divided into decade-long periods to test stationarity. Each decade period was saved as a separate signal. Consequently, data collections were created for temperature and pressure. The summary of the WD dataset, when each decade is considered as a separate signal, is provided in Table 6.

The WD summary is provided in the Table 6:

As seen in Figure 16, the boxplot of the daily average temperature signal(see Figure 16a) differs significantly from that of the raw temperature signal (see Figure 15a), exhibiting a broader range of normal values and notably fewer outliers. This boxplot shows that positive temperature values, considered outliers in the raw data, fall within the normal range when using daily averages. This suggests that excessively

Table 6: Weather Dataset (WD) summary

Property	Temperature	Pressure
amount of signals	6	6
amount of data points	21,910	21,858
mean	9.03	1016.96
std	7.68	8.63
min	-20.77	969.78
max	28.34	1046.80
lost signals?	yes	yes



(a) Boxplot of daily average temperature (b) Boxplot of daily average pressure

Figure 16: Boxplots for daily average Temperature and Pressure signals include the observations from 01.01.1961 to 31.12.2020 from the WD dataset. The horizontal orange line represents the median, the upper limit of the black box indicates Q3, and the lower limit shows Q1. The upper and lower whiskers point to borders for normal data. Points refer to potential anomalies.

high positive temperatures may be associated with instrument measurement errors rather than observed data.

In contrast, the boxplots for daily average pressure signal (see Figure 16b) and raw pressure signal (see Figure 15b) appear to be relatively similar, implying that potential anomalies in the pressure signal are related to the recorded pressure values at station 282. Boxplots for the remaining signals in the temperature and pressure collections of WD dataset are provided in the Appendix A.

To employ the TadGAN method on WD dataset, the daily average temperature and pressure signals were partitioned into the following datasets:

1. **Training**: This signal encompasses measurements taken between 01.01.1981, and 31.12.2000.

2. **Test 1**: Comprising measurements recorded from 01.01.1961, to 31.12.1980.

3. **Test 2**: This signal includes measurements gathered from 01.01.2001 to 31.12.2020.

The comprehensive summary of the training and test signals for both the pressure and temperature collections within the WD dataset utilised in TadGAN experiments is presented in Table 7. The summary shows that in the temperature signals, the mean value, minimum, and maximum values in Test 2 are higher than those in Test 1. This observation may indicate the presence of a trend in the signals of the Temperature collection. Simultaneously, the standard deviation in Test 2 is higher than that in Test 1, suggesting a narrower data range.

In the Pressure collection, the mean and maximum values exhibit negligible differences between Test 1 and Test 2, while the minimum value in Test 1 is six units lower than that in Test 2.

Table 7: Temperature and pressure collection signals using in TadGAN methods: Training dataset refers to temperature in 1981-2000, Test 1 refers to 1961-1980, Test 2 refers to 2001-2020

	Temperature			Pressure		
	Training	Test 1	Test 2	Training	Test 1	Test 2
amount of points	7,305	7,305	7,300	7,303	7,258	7,297
mean	9.0272	8.3259	9.7336	1,017.2918	1,016.4940	1,017.0909
std	7.7335	7.6551	7.5943	8.8468	78.5915	78.4392
min	-20.77	-20.54	-17.03	969.78	971.70	977.90
max	27.47	26.01	28.34	1,045.28	1,045.61	1,046.80
lost signals?	yes	no	no	yes	no	no

In this thesis, it is assumed that the training signals for temperature and pressure do not contain anomalies. Both training signals contained missing values, which were imputed using the interpolation method Bronstein et al. (2012). Therefore, the boxplots for the temperature and pressure Test signals will be shown in Figure 14. It illustrates that Test 1 (see Figure 14e) in Temperature collection may contain more anomalies than Test 2 (see Figure 14f). For the test signals in the Pressure Collection, constructing a similar hypothesis based on the boxplots is challenging. Both Test 1 (see Figure 14g) and Test 2 (see Figure 14h) appear to contain an equal number of anomalies. The only hypothesis that can be formulated is that the Test 1 and Test 2 signals from the Pressure collection may contain more anomalies than the Test 1 and Test 2 signals from Temperature collection data.

The application of the ARIMA Box et al. (2015) method poses greater challenges. Firstly, as described in Chapter 3.2, ARIMA is a prediction-based method. Consequently, it is difficult to use the same training and test signals as those used for the TadGAN method, as predicting the past values using the ARIMA method has

lower performance. Moreover, statistical methods, including ARIMA, are not a good choice for forecasting many steps ahead Chan (2004).

The complete daily average temperature signal comprises 21, 910 data points, while the entire pressure signal contains 21, 858 points. In such cases, parameter selection for ARIMA and obtaining accurate forecasts becomes challenging. To address this, the following averaging approach was applied to prepare training and test signals for anomaly detection based on the ARIMA method.

The monthly average temperature was computed for the daily average temperature signal, and the monthly average pressure was calculated for the daily average pressure signal. This allowed the selection of appropriate ARIMA model parameters.

The acquired signal was divided into training and test signals. The Training signals were utilised for the parameter selection of the ARIMA model, while the Test signal was employed for signal prediction and anomaly detection. The Training signal comprises the average monthly temperature (or monthly pressure respective to the collection) from 01.1961 to 12.2000, and the Test signal contains the average monthly temperature (or monthly pressure respective to the collection) from 01.2001 to 12.2020. The Table 8 presents a summary of these signals.

Table 8: The signals used in ARIMA methods for the temperature and pressure collection consist of three parts: the Training signal covering the years 01.1961-12.2000, the Test signal for 01.2001-12.2020, and the signal spanning the entire observation period from 01.1961 to 12.2020. These signals contain average monthly temperature and pressure values.

	Temperature			Pressure		
	Training	Test	Signal	Training	Test	Signal
amount of points	576	480	720	576	480	720
mean	8.8159	8.635	8.9879	1,016.9405	1016.8826	1,016.95
std	6.929	6.9196	6.9059	4.0356	4.0523	4.037
min	-7.3474	-7.3474	-7.3474	1004.0196	1004.0196	1,004.01
max	22.0783	22.5541	23.0784	1033.079	1033.079	1033.07
lost_signal?	no	no	no	no	no	no

Subsequently, the stationarity of each signal in the temperature and pressure collections was analysed using methods and algorithms described in Chapter 3.1.

Based on the combinations of ADF and KPSS tests for both regression parameters *Constant* and *Constant and trend* in a test procedure without any limitation on the maximum number of lags, all signals in the WD dataset are classified as TS-Stationary signals, indicating the presence of a stable trend. This aligns with the hypothesis of a global temperature increase.

On the other hand, when using only the ADF Test with all possible regression parameters, namely constant (c), constant and linear trend (ct), constant and quadratic trend (ctt), and without trend (n), without any limitation on the maximum number of lags, some differences arise for some signals from Temperature collection.

Table 9 indicates that under the regression parameter *ctt* both the Training and Test datasets created for applying the ARIMA method are classified as non-stationary. The same applies to the three decades signals from the Temperature collection. This suggests that the temperature data may have a linear trend, or that further limitations on the *ctt* parameter of linear regression are needed for the Temperature collection of the WD Dataset.

Table 9: Testing for stationarity (signals without a unit root) in each signal from the Temperature and Pressure collection of the WD dataset was conducted using the ADF test with different regression models, including constant (c), constant with linear trend (ct), constant with linear and quadratic trend (ctt), and without constant and trend (n) regression models in the NAB Dataset. The analysis involved examining an unlimited number of lags. S indicates stationarity, while N indicates non-stationarity.

	c		ct		ctt		n	
	S.	N.	S.	N.	S.	N.	S.	N.
(T)Decades	6	-	6	-	3	3	6	-
(T)TadGAN (training, tests)	1	-	1	-	1	-	1	-
(T)ARIMA (training, test)	1	-	1	-	-	1	1	-
(P)Decades	6	-	6	-	6	-	6	-
(P)TadGAN (training, tests)	1	-	1	-	1	-	1	-
(P)ARIMA (training, test)	1	-	1	-	1	-	1	-

The results of applying the ARIMA and TadGAN methods on the WD dataset will be presented in Chapter 5.3.

5 Results

In this chapter, the experimental results of applying selected datasets, described in Chapter 4, to the methods outlined in Chapter 3, are presented. Chapter 5.1 describes the experimental setup for the selected state-of-the-art GAN-based methods: TadGAN Geiger et al. (2020) and TAnoGAN Bashar and Nayak (2020), as well as the classical statistical ARIMA Box et al. (2015) method. Chapter 5.2 presents the replication results of the selected GAN-based methods on the labelled NAB dataset and demonstrates the performance of these two selected GAN-based methods, depending on the stationarity types of signals in the NAB dataset. Chapter 5.3 shows the results of applying the ARIMA and TadGAN methods to temperature and pressure signals from the WD Dataset.

5.1 Experimental Setup

5.1.1 Experimental Setup for Reproduction of Selected State-of-the-Art GAN-based Methods

TAnoGAN experimental setup [11]

1. **Training and test datasets**. The model is trained separately for each signal from the NAB dataset. Each signal is divided as follows: the initial 70% of the signal is utilised for model training, and the remaining 30% is reserved for signal reconstruction using the trained model and anomaly detection.

2. **Data Normalisation** Signal is normalised using the *StandardScaler* from the *sklearn* library. Normalisation is performed so that the mean value of each data column becomes equal to 0, and the standard deviation becomes equal to 1.

3. **Sequence generation for training**: o generate a sequence from real data within the domain $\mathbb{X}$, a sliding window with a window size of $s_w = 60$ and a step size of $s_t = 1$ was applied. For generating noise from a latent space of dimension $noise\_dim = 100$ (domain $\mathbb{Z}$), samples were drawn from a standard normal distribution $\mathcal{N}(0, 1)$ with a batch size of 8 and a time sequence length (seq_len) of 16.

4. **TAnoGAN training parameters**: The following parameters were used for training the algorithm: epoch = 20, workers = 4, batch size = 32, optimizer: *Adam* with a learning rate of 0.0002. Real data from domain $\mathbb{X}$ is labelled as 1, while generated data from domain $\mathbb{Z}$ is labelled as 0.

[11] Based on Bashar and Nayak (2020) and GitHub Repository: `https://github.com/mdabashar/TAnoGAN/blob/master/TAnoGAN_Pytorch.ipynb`

5. **Inverse Mapping**: Initial parameters for inferse mapping: workers = 1, batch size = 1, optimizer for **z**: *Adam* with a learning rate of 0.01, anomaly score parameter $\gamma = 0.1$

6. **TAnoGAN testing and anomaly detection** A sliding window with a window size of $s_w = 60$ and a step size of $s_t = 30$ is used to generate sequences from the test dataset. Subsequently, a threshold is determined, and the loss and anomaly functions were plotted. Bashar and Nayak (2020) did not provide specific mathematical assumptions for determining thresholds but mentioned that they were chosen manually. In this study, a similar approach was adopted, and thresholds for each signal were individually selected as the minimum value that ensures at least one TP value, as demonstrated in Figure 17, enabling the calculation of accuracy, precision, recall, and F1-Score (these metrics were described in detail in Chapter 3.4). The mean value of all thresholds for each collection of the NAB dataset is provided in Table 10.

Table 10: Mean ± standard deviation of thresholds used for various collections in the NAB dataset to apply the TAnoGAN method.

	Art	AdEx	AWS	Known	Traf	Tweets
Threshold	$6 \pm 3,46$	6 ± 3	$7,77 \pm 0,52$	$6,48 \pm 3,19$	$7,15 \pm 1,89$	4 ± 0

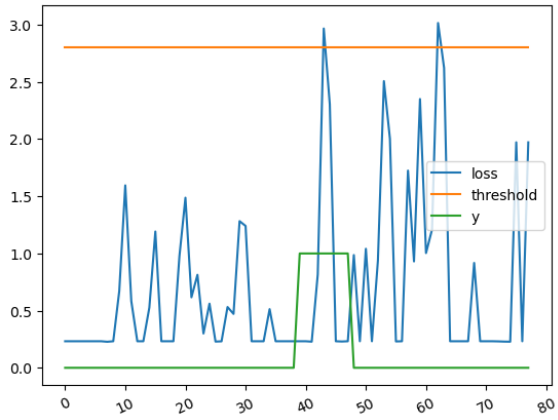


Figure 17: The TAnoGAN error plot of signal *ec2_disk_write_bytes_1ef3de* from AWS collection. The blue line represents the reconstruction loss. The orange line indicates the threshold. The green line depicts the actual anomalies.

TadGAN experimental setup [12]

[12]Reproduction experiments for TadGAN Method are based on the parameters presented in the pipelines setup for Orion-ml library. https://github.com/sintel-dev/Orion/tree/master/orion/pipelines/verified/tadgan

1. **Training and test dataset**: TadGAN is trained on each signal from the NAB dataset, and the trained model is then used to reconstruct the same signal and calculate the reconstruction error. To adjust the intervals between measurements within the signal, it was essential to specify the recording interval presumed (or indicated in the dataset's annotation) for the signal values. Row *interval* in Table 11 shows the interval for each collection in the NAB dataset. For *Known* collection these intervals were not standardised and were as follows: nyx_taxi = 1800, *ambicient_temperature_system_failure* = 600, *cpu_utilization_asg_misconfiguration* = 300, *ec2_request_latency_system_failure* = 300, *rogue_agent_key_hold* = 300, *rogue_agent_key_updown* = 300

   As TadGAN utilises numerical timestamp values (i.e., the number of seconds since the computer's reference time), it is imperative that the intervals are specified in seconds.

Table 11: Default experimental setups for the TadGAN method vary across different collections in the NAB and Yahoo datasets. This information is derived from the *Orion-ml* library.

	NAB					
	Art	AdEx	AWS	Known	Traf	Tweets
interval	600	3600	600	*	600	600
error_type	dtw	dtw	dtw	dtw	dtw	dtw
combination	mult	sum	mult	mut	sum	sum

2. **Missing value imputation**: Any missing values in sugnal were filled with the mean value of the entire signal using the *SimpleImputer* method from the *sklearn* library.

3. **Normalisation of measurements**: The values were normalised to the interval $[-1, 1]$ using the *MinMaxScaler* method from *sklearn* library.

4. **Sequence generation for training and test**: For training the algorithm, a sliding window of length $s_w = 100$ with a step size of $s_t = 1$ was utilised to generate sequences from the real signal. These sequences were then used for training the model and detecting anomalies.

5. **TadGAN training parameters**: For TadGAN training, the following parameters were utilised: epoch = 35, dimension of the latent space($dim(Z)$) = 20, batch size = 64, optimizer: *Adam* with a learning rate of 0.0005.

6. **Reconstruction of timesequence**: After training, the TadGAN model reconstructed a value $\mathbf{X}^{(i)}$ for each sliding window separately (see Chapter 3.3.3 or Geiger et al. (2020)). To obtain the predicted signal $\hat{\mathbf{X}}^{(i)}$, the *mean* aggregation method was employed.

7. **Anomaly detection**: The difference between the real and reconstructed signals is used to detect anomalies. In Table 11, the rows *error_type* and *combination* are presented default settings for anomaly detection using the *Orion-ml* library. *Error_type* refers to the error mechanism used to compute the difference between the reconstructed and authentic signals, while *combination* specifies whether to use multiplication (mult) or convex combination (sum) with the parameter $\alpha = 0.5$ for the computed reconstructed error and C_x, which represents the Critic error to compute anomaly score. Chapter 5.2 provides reproduction results for the following parameters: error_type = dtw,combination = mult. The results for default parameters for all signals in the NAB dataset are provided in the GitHub repository.[13]

   Unlike the TAnoGAN method, which employs a fixed threshold, TadGAN utilizes a window-based adaptive threshold to determine anomalies. The parameters governing the window approach are as follows: $window\_size\_portion = 0.33$ of $window\_size$, $window\_step\_size\_portion = 0.1$ of $window\_size$, where $window\_size = 100$.

5.1.2 Experimental Setup for Anomaly Detection in Weather Dataset

ARIMA For the Temperature and Pressure collections from the WD dataset, the following parameters were applied for the ARIMA Box et al. (2015) method.

1. **Training and test signals**: Two training signals (Training), one for temperature and one for pressure, contained average monthly measurements from 01.1961 to 12.2000 were used as training data to fit the ARIMA parameters and to train the ARIMA model with the chosen one. Two test signals (Test), one for temperature and one for pressure, contained average monthly measurements from 01.2001 to 12.2020 (see Chapter 4.4 for a description of each of the signals) were used as testing data with fitted ARIMA model to check the quality of prediction of the ARIMA model and detect anomalies.

2. **Missing value imputation**: Any missing values were filled with the mean value of the entire signal using the *SimpleImputer* method from the *sklearn* library.

3. **Choose the best ARIMA model**: As mentioned in the corresponding Chapter 3.2, the $ARIMA(p, d, q)$ model requires empirical determination of its parameters. To conduct the parameter tuning experiment, it was necessary to establish a criterion for selecting the best model. The standard Akaike Information Criterion (AIC) Akaike (1974) was used in this study. To facilitate the model parameter selection process, the *auto.Arima()* function from the *pmdarima* library was employed.[14] The *auto.Arima()* parameters for the model fitting were set as follows:

[13] GitHub
[14] https://pypi.org/project/pmdarima/

- $m = 12$ implies that the model interprets Test signals as referring to monthly data.
- seasonal = $\{True, False\}$, because the both options were tested.
- ADF test were used to find optimal differencing parameter d
- $p, q = 0, ..., 12$

The ARIMA model with the lowest AIC was chosen as the best model and used for the following predictions. The best models for temperature and pressure monthly average signals are shown in Table 12. The least complex SARIMA model was selected for the temperature signal, while the model with the lower AIC criterion was chosen for the pressure signal.

Table 12: Hyperparameters for an optimal ARIMA (p,d,q) or SARIMA (p,d,q)(P,D,Q)[s] model, based on AIC criteria, for analyzing test signals derived from monthly average temperature and pressure data collections in the WD Dataset. This assessment encompasses two possibilities: the seasonal model SARIMA ((Seasonal) and the non-seasonal time series model ARIMA (Non-seasonal). **Bold** refers to the chosen model.

	Temperature		Pressure	
	Model	AIC	Model	AIC
Seasonal	**SARIMA(0,0,0) (1,0,2)[12]**	2,062.570	**SARIMA(0,0,2) (0,0,1)[12]**	2,684.906
Non-seasonal	ARIMA(12,0,4)	2,022.570	ARIMA(2,0,0)	2,698.657

4. **Test and anomaly detection**: For prediction, a one-step-ahead forecast with a chosen SARIMA model was used. To detect anomalies, the threshold method was employed. The threshold boundary was determined as the sum of the mean squared error and the square of the standard deviation error, scaled by a coefficient denoted as α, which signifies the relative importance of the standard deviation error:

$$threshold = mean\_squared\_error + (\alpha * std\_squared\_error) \quad (33)$$

In the experiments conducted for this study $\alpha = 2$. A measurement is considered anomalous if the difference between the true and predicted values exceeds the threshold value.

TadGAN The following experiment parameters were applied to the temperature and pressure signals from the WD dataset:

1. **Training dataset**: Two signals, one for temperature and one for pressure, containing average daily measurements from 01.01.1981 to 31.12.2000 were used as training data for the TadGAN method.

2. **Test datasets**: The data for detecting anomalous regions consists of two signals for Temperature and Pressure collections:

   - Test 1: This signal includes daily measurements from 01.01.1961 to 31.12.1980.
   - Test 2: This signal encompasses daily measurements from 01.01.2002 to 31.12.2020.

   The detailed description of Training and Test 1 and Test 2 signals for Temperature and Pressure collections of WD dataset was provided in Chapter 4.4.

3. **Date Transformation to TadGAN Format**: To utilise the TadGAN algorithm, it was necessary to transform the date into the TadGAN format, following the procedure, described in *Orion-ml* library to the data transformation for the NASA dataset[15]. For this purpose, the *timestamp* column was modified. The first row of the Training and Test 1 signals was set with a value of *347151600 (1981-01-01T00:00:00)*, and then the timestamp was incremented by 86400 seconds (24 hours) for each subsequent row as measurements were provided once a day. Using real timestamps for Test 1 signal resulted in negative values in timestamps and errors in reconstruction. The first row of the Test 2 signal was set with a value of *978303600 (2001-01-01T00:00:00)*, and then the timestamp was incremented by 86400 seconds (24 hours) for each subsequent row due to daily measurements.

4. The following steps and parameters are the same as on pp.2-5 in the experimental setup of the TadGAN method for the NAB dataset.

5. **Anomaly detection**: For TadGAN testing, two datasets, namely Test 1 and Test 2, were used. Anomalies were detected using multiplication or convex combination of all different methods for computing a reconstruction error (pointwise difference, area difference, and DTW) and C_x Critic error. The window approach parameters for adaptive threshold were defined as follows: window_size_portion = 0.005, window_step_size_portion = 0.1.

Information regarding the number of detected anomalies using the chosen SARIMA model and the results of anomaly detection using TadGAN will be presented in Chapter 5.3.

[15] https://github.com/sintel-dev/Orion/blob/master/tutorials/Convert_NASA_Data_to_Orion_Format.ipynb

5.2 Reproduction Results on Selected State-of-the-Art Methods

5.2.1 Comparative Analysis of Experimental Results Against Baseline Performance

The first research question of this study refers to reproducing selected state-of-the-art GAN-based methods.

Two methods based on the principles of GAN, as proposed in the paper by Goodfellow et al. (2014), were selected as state-of-the-art methods for reproducing the results provided in the original papers. The first approach, TAnoGAN Bashar and Nayak (2020), is a classical GAN, while the second, TadGAN Geiger et al. (2020), offers a novel implementation of the GAN method for anomaly detection in signals. Both of these approaches have been thoroughly described in Chapter 3.

As the baseline method, the ARIMA Box et al. (2015) method was chosen. It is important to note that the ARIMA method was not reproduced in this study due to a lack of sufficient information regarding its parameters, used in Geiger et al. (2020). Additionally, Bashar and Nayak (2020) did not employ classical statistical methods as a baseline. To assess the quality of the results, F_1-Score values, as indicated in the paper by Geiger et al. (2020), were used.

The NAB dataset, which comprises 53 time series divided into 6 different collections, was utilised to reproduce the results. A detailed description of the NAB dataset was presented in Chapter 4.3.

As a method for computation of anomaly score, the $DTW \times critic$ combination from the original paper was chosen to check the performance of the TadGAN method, as it demonstrated the best results among all the datasets in the article of Geiger et al. (2020). For TAnoGAN, thresholds that deviated significantly from the most commonly used thresholds within each collection were discarded.

A summary of the reproduction results according to F_1-Score is presented in Table 13. It contains the best reproduction results obtained during the work (our b), the average results within all signals within each Collection (our m), and results from papers (Bashar and Nayak (2020) for TAnoGAN and Geiger et al. (2020) for TadGAN (paper). Additionally, Geiger et al. (2020) did not use the Known collection of the NAB dataset. Furthermore, Bashar and Nayak (2020) did not provide any summary information within the Collections of NAB dataset. The absence of information in the table is denoted by the symbol '−'. As seen in Table 13, the original results are partially reproducible. The best results outperform the results provided by Geiger et al. (2020) in 4 out of 5 collections but not in the AdEx collection. The mean results differ noticeably, with only a small deviation in the AWS collection (0.012). The Art, Traf, and Tweets collections show stronger deviations (0.036, 0.095, 0.035). However, the strongest deviation is observed in the AdEx Collection, where the difference between the paper's results and the computed mean value is 0.326.

Table 13: F_1-Score of chosen State-of-the-Art methods in the NAB Dataset. *Our b* refers to the best reproduction results within the collection, *our m* refers to mean results within all signals of collection while *paper* refers to results from articles: Bashar and Nayak (2020), Geiger et al. (2020), when the results are available. ARIMA results for each collection refer to results from Geiger et al. (2020) and were not reproduced in this thesis.

Data	ARIMA	TAnoGAN			TadGAN		
	paper	our b	our m	paper	our b	our m	paper
Art	0.353	0.842	0.551	-	0.724	0.631	0.667
AdEx	0.583	0.545	0.357	-	0.433	0.341	0.667
AWS	0.518	0.842	0.489	-	0.907	0.598	0.61
Known	-	0.7	0.490	-	0.539	0.354	-
Traf	0.571	0.769	0.359	-	0.522	0.360	0.455
Tweets	0.567	0.608	0.422	-	0.866	0.570	0.605

Although the authors of the TAnoGAN method did not provide numerical summaries for collections in NAB dataset, based on the plots they provided in the paper Bashar and Nayak (2020), it can be concluded that the TAnoGAN F_1-Score falls within the range $[0.4, 0.9]$ for each signal in the NAB dataset. The reproduced results within this study for each signal in the NAB dataset fall within the range $[0.22, 0.93]$. There are signals that deviate from the expected results. The signals within the expected F_1-Score range are mostly from the AdEx, Traf, and Tweets collections.

Based on the results of Geiger et al. (2020), TadGAN should outperform ARIMA results in 4 out of 5 Collections under the study (without Known Collection). In the reproduced results, TadGAN outperforms ARIMA in 3 out of 5 collections, regardless of whether the mean or the best results were chosen. In the reproduced results, TAnoGAN outperforms ARIMA in 5 out of 5 collections when the best results were chosen and in 1 out of 5 collections (Art) when the mean method was chosen.

It is also important to note that, based on the results in Table 13, the best results obtained by the TAnoGAN method are definitely greater than those achieved by TadGAN in 4 out of 6 collections. However, the average results of TAnoGAN are better than those of TadGAN in 2 of the 6 collections.

Table 14 summarises the mean accuracy, precision, recall, and F_1-Score for all signals in the NAB dataset, based on the results obtained through the experiments conducted for this study (see Chapter 3.4 for description of statistical measurements). It shows that TadGAN outperforms TAnoGAN in 5 out of 6 collections in terms of accuracy, and in the Known collection, it outperforms TAnoGAN. This indicates that TadGAN better identifies anomalous measurements in signals of the NAB dataset.

Moreover, TadGAN demonstrates better results in 4 out of 6 collections for precision, suggesting that the TadGAN method is more capable of correctly identifying real anomalies (TP) among all points classified as anomalies ($TP + FP$). On the other hand, TAnoGAN outperformed TadGAN in 5 out of 6 collections of the NAB dataset based on the recall measure. This suggests that TAnoGAN better captures actual anomalies among all the true anomalies present in the signals compared to TadGAN in those cases. Regarding the F_1-Score, TadGAN outperforms TAnoGAN in 4 out of 6 collections in NAB.

Table 14: Accuracy, precision, recall and F_1-Score of chosen State-of-the-Art GAN methods in the NAB Dataset. The results for each of the six collections are presented as the mean of individual results, rounded to three decimal places.

Data	TAnoGAN				TadGAN			
	Ac	Pr	Re	F_1	Ac	Pr	Re	F_1
Art	0.751	0.412	**1.000**	0.551	**0.862**	**0.675**	0.450	**0.631**
AdEx	0.722	**0.363**	**0.667**	**0.357**	**0.738**	0.248	0.588	0.341
AWS	0.747	0.432	**0.728**	0.489	**0.853**	**0.568**	0.568	**0.598**
Known	**0.731**	0.485	**0.756**	**0.490**	0.715	**0.520**	0.216	0.354
Traf	0.739	**0.420**	0.604	0.359	**0.803**	0.276	**0.670**	**0.360**
Tweets	0.676	0.442	**0.771**	0.422	**0.887**	**0.491**	0.378	**0.570**

5.2.2 Performance Analysis of Selected Anomaly Detection Methods Based on Signal Stationarity

The second research question of this thesis pertains to whether the results of GAN-based methods depend on signal properties. To investigate this, a combination of KPSS and ADF tests was used (refer to the test descriptions in Chapter 3.1 and the description of the NAB dataset dependent on stationarity in Chapter 4.3) to examine how stationarity of signal influenced the performance of the TadGAN and TAnoGAN methods.

Table 15 presents the mean values for precision, recall, and F_1-Score for TS-Stationary, Stationary, and DS-Stationary signals with respect to the combination of ADF and KPSS tests with the Constant (c) and Constant and Trend (ct) linear regression parameters for a TadGAN method.

For the regression parameter c, the mean value of prediction for Stationary signals (0.537) outperforms the mean value of DS-Stationary signals (0.488) and TS-Stationary signals (0.404) in terms of precision. However, for recall and F_1-Score, the mean value for DS-Stationary signals outperforms the mean value for both TS-Stationary and Stationary signals.

Table 15: Precision, recall, and F_1-Score of TadGAN method on the NAB Dataset, considering different types of stationarity in signals based on a combination of ADF and KPSS tests with a constant (c) or constant and trend (ct) regression parameter. The results for each of the six collections are presented as the mean of individual results, rounded to three decimal places.

Collection	TS-Stationary			Stationary			DS-Stationary		
	Pr	Re	F_1	Pr	Re	F_1	Pr	Re	F_1
Art (c)	0.631	0.412	0.603	**0.895**	**0.637**	**0.744**	-	-	-
AdEx (c)	0.250	**0.951**	0.395	0.225	0.429	0.292	**0.313**	0.703	**0.433**
AWS (c)	0.415	0.493	0.479	**0.853**	**0.786**	**0.772**	0.662	0.556	0.675
Known (c)	0.316	**0.266**	0.288	**0.657**	0.183	**0.420**	-	-	-
Traf (c)	0.230	0.523	0.292	**0.311**	**0.780**	**0.411**	-	-	-
Tweets (c)	**0.581**	**0.436**	**0.576**	0.283	0.244	0.392	-	-	-
Art (ct)	0.631	0.412	0.603	**0.895**	**0.637**	**0.744**	-	-	-
AdEx (ct)	0.268	**0.805**	0.397	0.195	0.315	0.238	**0.313**	0.703	**0.433**
AWS (ct)	0.246	0.389	0.279	0.539	**0.567**	0.585	**0.662**	0.556	**0.675**
Known (ct)	-	-	-	0.520	0.216	0.354	-	-	-
Traf (ct)	0.238	0.583	0.314	**0.291**	**0.704**	**0.379**	-	-	-
Tweets (ct)	0.472	0.351	**0.557**	**0.536**	**0.442**	0.483	-	-	-
mean (c)	0.404	0.513	0.439	**0.537**	0.510	0.505	0.488	**0.630**	**0.554**
mean (ct)	0.371	0.508	0.430	**0.496**	0.480	0.464	0.488	**0.630**	**0.554**

For the regression parameter ct, the mean value of performance for Stationary signals (0.496) outperforms the mean value of DS-Stationary signals (0.488) and TS-Stationary signals (0.371) in terms of precision. Similarly, for recall and F_1-Score, the mean value for DS-Stationary signals outperforms the mean value for both TS-Stationary and Stationary signals.

Table 15 shows trend where the average performance of TS-Stationary, Stationary, and DS-Stationary signals classified with a combination of ADF and KPSS tests with the regression parameter c outperforms the performance of TS-Stationary, Stationary, and DS-Stationary signals classified with ct regression parameters.

Within the results of each collection in the NAB dataset, Table 15 shows that, for the regression parameter c, the Stationary signals exhibit the best precision and F_1-Score results in 2 collections: Art and Traf, and the best recall results in 2 collections: Art and Traf. TS-Stationary signals display the best precision and the best F_1-Score in the Tweets collection, as well as the best recall in 2 collections: AdEx and Tweets.

Conversely, in the AWS collection, the best results in precision, recall, and F_1-score are demonstrated by DS-Stationary signals. Under this regression parameter, the performance of TS-Stationary and Stationary signals outperform the results of DS-Stationary signals within 5 of the 6 collections of the NAB dataset.

A similar tendency was observed for the regression parameter ct. The Stationary signals exhibit the best precision and F_1-Score results in 3 collections: Art, Traf and Tweets, and the best recall results in 4 collections: Art, AWS, Traf and Tweets. TS-Stationary signals display the best F_1-Score in the Tweets collection and the best recall in AdEx. Conversely, in the AWS and AdEx collections, the best results in precision and F_1-score are demonstrated by DS-Stationary signals. Under this regression parameter, the precision of TS-Stationary and Stationary signals outperforms the results of DS-Stationary signals within all signals of the NAB dataset, although DS-Stationary signals show the best precision and F_1-score results in two collections of NAB.

Indeed, regardless of the regression parameter chosen for stationarity tests, there is a tendency in TadGAN where the method's application yields better results for stationary signals compared to DS-Stationary (or non-stationary) signals.

Table 16 presents the mean values for precision, recall, and F_1-Score for TS-Stationary, Stationary, and DS-Stationary signals with respect to the combination of ADF and KPSS tests with the Constant (c) and Constant and Trend (ct) linear regression parameters for a TAnoGAN method. Mean was computed within all signals of each collection without any threshold constraints.

For the regression parameter c, the mean value of precision for Stationary signals (0.485) outperforms the mean value of TS-Stationary signals (0.449) and DS-Stationary signals (0.286). The same trend is observed for recall and F_1-Score: the mean value of Stationary signals outperforms the mean value of TS-Stationary signals, which, in turn, outperforms the mean value of DS-Stationary signals.

For the regression parameter ct, the mean value of precision for TS-Stationary signals (0.522) outperforms the mean value of Stationary signals (0.440) and DS-Stationary signals (0.287). The mean value of the F_1-Score demonstrates a similar trend. However, for the recall parameter, the mean value for Stationary signals (0.792) is greater than the mean value for TS-Stationary signals (0.732), which also outperforms the mean value for DS-Stationary signals (0.577).

Within the results of each collection in the NAB dataset, Table 16 shows that, for the regression parameter c, the Stationary signals exhibit the best precision results in 2 collections: Known and Traf, the best recall results in 2 collections: AWS and Tweets and the best F_1-Score in 2 collections: Known and Tweets. TS-Stationary signals display the best precision in 4 collections: Art, AdEx, AWS, Tweets, the best recall in 2 collections: Known and Traf and the best F_1-Score in 4 collections: Art, AdEx, AWS, Traf. In the AWS collection, the best result in recall is demonstrated by DS-Stationary signals. Under this regression parameter, the performance of TS-Stationary and Stationary signals outperform the results of DS-Stationary signals within 5 of the 6 collections of the NAB dataset.

Table 16: Precision, recall, and F_1-score of TAnoGAN method on the NAB Dataset, considering different types of stationarity in signals based on a combination of ADF and KPSS tests with a constant (c) or constant and trend (ct) regression parameter. The results for each of the six collections are presented as the mean of individual results, rounded to three decimal places.

Collection	TS-Stationary			Stationary			DS-Stationary		
	Pr	Re	F_1	Pr	Re	F_1	Pr	Re	F_1
Art (c)	**0.495**	**1.000**	**0.624**	0.444	**1.000**	0.615	-	-	-
AdEx (c)	**0.500**	0.333	**0.400**	0.319	0.611	0.363	0.220	**0.750**	0.277
AWS (c)	**0.658**	0.779	**0.658**	0.591	**0.833**	0.580	0.351	0.655	0.452
Known (c)	0.159	**0.800**	0.261	**0.668**	0.662	**0.584**	-	-	-
Traf(c)	0.419	**0.736**	**0.526**	**0.489**	0.479	0.332	-	-	-
Tweets (c)	**0.461**	0.717	0.374	0.396	**0.931**	**0.534**	-	-	-
Art (ct)	**0.495**	**1.000**	**0.624**	0.444	**1.000**	0.615	-	-	-
AdEx (ct)	**0.500**	0.333	**0.400**	0.225	**0.833**	0.311	0.222	0.500	0.308
AWS (ct)	**0.717**	0.611	**0.647**	0.611	**0.863**	0.633	0.351	0.655	0.452
Known (ct)	-	-	-	0.523	0.701	0.492	-	-	-
Traf (ct)	0.424	**1.000**	**0.566**	**0.473**	0.425	0.354	-	-	-
Tweets (ct)	**0.474**	0.717	0.392	0.366	**0.931**	**0.492**	-	-	-
mean (c)	0.449	0.728	0.474	**0.485**	**0.753**	**0.501**	0.286	0.702	0.364
mean (ct)	**0.522**	0.732	**0.526**	0.440	**0.792**	0.483	0.287	0.577	0.380

For the regression parameter ct, the Stationary signals exhibit the best precision results in Traf collection, the best recall results in 3 collections: AdEx, AWS and Tweets and the best F_1-Score results in Tweets collection. TS-Stationary signals display the best precision in 4 collections: Art, AdEx, AWS, Tweets, the best recall in 2 collections: Art and AWS. Under this regression parameter, DS-Stationary signals did not show the best results in any of the detected evaluation metrics.

Based on the results provided in Table 15 and Table 16, it is evident that the performance of TadGAN and the performance of TAnoGAN differ for signals with different types of stationarity. However, the TAnoGAN method exhibits the worst results for DS-Stationary signals. It simultaneously shows a different best F_1-Score for TS-Stationary and Stationary signals, depending on the regression parameters of ADF and KPSS tests. At the same time, TadGAN demonstrates more consistent results. In the case of choosing the regression parameter ct the results of all selected metrics were worse than when choosing the regression parameter c for ADF and KPSS tests. Additionally, it can be noticed that for the TadGAN method, precision

and F_1-Score measurements were directly related to each other: the best precision result for signals with a certain type of stationarity also indicated the best F_1-Score result. The results obtained when applying TAnoGAN do not align, meaning that the best precision does not tend to yield better recall or F_1-Score.

Based on the description provided in Chapter 5.2, it is evident that TadGAN consistently outperforms TAnoGAN across all four statistical metrics: accuracy, precision, recall, F_1-Score. Additionally, it was demonstrated in this chapter that TadGAN yields stable results for signals with different types of stationarity. As a result, for the application on the unlabeled WD dataset, the TadGAN algorithm was selected as the most suitable among the chosen state-of-the-art methods.

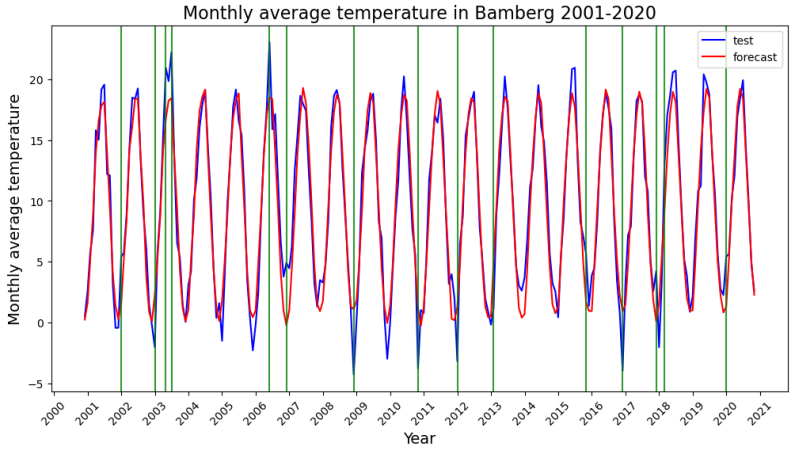
5.3 Results of Anomaly Detection in Weather Dataset Using ARIMA and TadGAN

5.3.1 Anomaly Detection Results Using ARIMA

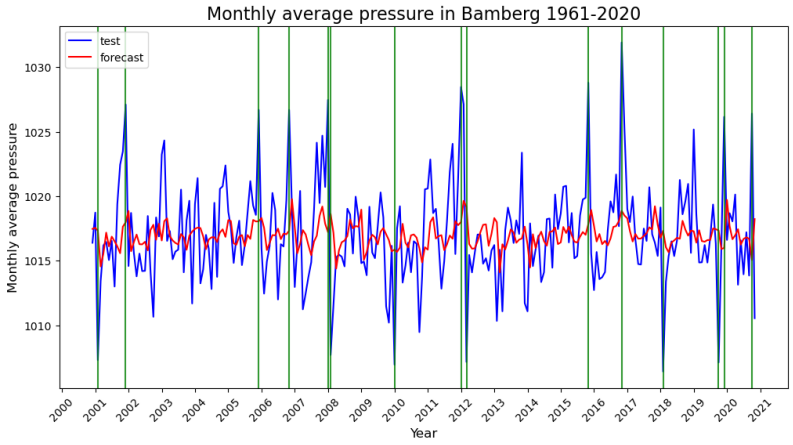
The third research question pertains to detecting potential anomalies in the unlabeled WD dataset (see the description in Chapter 4.4). To detect anomalies using ARIMA, the experiment was conducted with the parameters presented as described in Chapter 5.1.2. The results are displayed in the figure 18. For temperature and pressure Test signals, containing monthly average temperature in 01.2001-12.2020 chosen SARIMA models detected 15 anomaly months. Figure 18a reveals that there were detected only 5 years within the 20-year period when no anomaly temperature months were detected, specifically in the years 2001, 2004, 2005, 2007, and 2009. Most anomalies were detected during the winter months (9) when the average monthly temperature was lower than predicted by the model. The summer anomalies referred to lower temperatures than what was detected. Figure 18b demonstrates a similar trend for the Test signal from the Pressure collection: 7 anomalous months are associated with winter months. The number of years with no detected anomalous months remains the same. In 5 years—2002, 2003, 2004, 2013, and 2014—no anomalous pressure measurements were identified using the selected ARIMA model. Only one anomaly month: 01.2012 was the same for both, pressure and temperature Test signals of WD dataset.

5.3.2 Anomaly Detection Results Using TadGAN

For anomaly detection in the WD Dataset using TadGAN, the daily average Training, Test 1 and Test 2 signals for Temperature and Pressure collection were used. The experimental setup, which is described in detail in Chapter 5.1, involved examining various combinations of reconstructed error metrics (pointwise difference, area difference, or DTW) and C_x critic error (critic). These combinations were combined using multiplication ($\times$) or convex combination ($+$). Each detected anomalous region spans 36 days. A summary of the results is presented in Table 17.



(a) Anomalie in temperature Test signal. Method: SARIMA(0,0,0)(1,0,2)[12]



(b) Anomalie in pressure Test signal. Method: SARIMA(0,0,3)(0,0,1)[12]

Figure 18: Results of the applied SARIMA methods on monthly average measurements from 01.2001 to 12.2020 temperature and pressure test signals of WD Dataset. The blue line represents the true test signal, the red line represents the reconstructed signal, and the green vertical lines indicate anomalies.

Table 17: The number of group anomalies detected by TadGAN in the temperature and pressure datasets for Test 1 (01.01.1961-31.12.1980) and Test 2 (01.01.2000-31.12.2020) varies depending on the approach used for computing the reconstruction error. The **Bold** indicates the most detected anomalies within the signal. The *Italicised* corresponds to baseline settings for anomaly detection with TadGAN method that was employed for reconstructing results on the labelled NAB dataset.

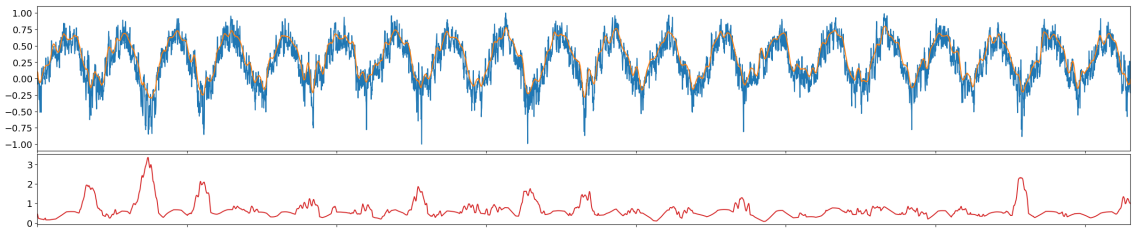
Variation	Temperature		Pressure	
	Test 1	Test 2	Test 1	Test 2
critic + point	13	17	21	19
critic × point	**24**	**21**	**24**	**26**
critic + area	11	11	13	9
critic × area	16	14	20	15
critic + DTW	7	7	12	14
critic × DTW	*12*	*17*	*14*	***26***

The number of anomalies detected using the multiplication of the reconstruction error and the C_x Critic error outperforms the number of detected anomalies using a convex combination for each type of reconstruction error across all Test signals in the TadGAN method. In Figure 19, a real and reconstructed signal from Test 1 of the Temperature collection is shown, along with associated reconstruction error plots for the convex combination (see Figure 19a) and the multiplication (see Figure 19b) of both scores. The maximum value of the convex combination of errors is 3, whereas for multiplication, it reaches 15. However, both combination methods yield the same number of peaks in the error plots. The error plot of the convex combination shows more local turning points, while the error plot of multiplication appears more stable, with fewer local extremes. The plots for other combinations of reconstruction error and critic C_x are provided in Appendix A.

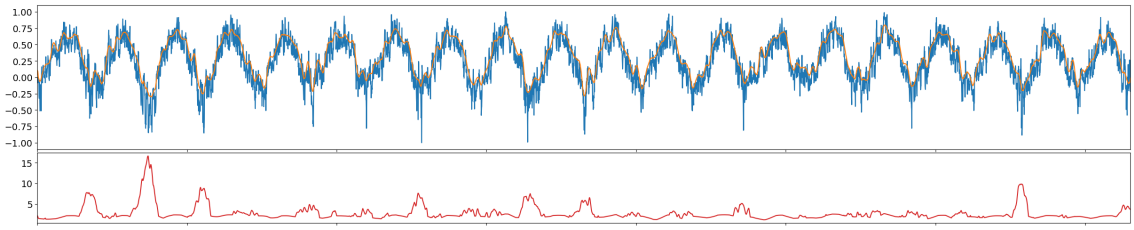
Table 17 shows that the majority of anomalies were detected using multiplication of point-wise difference and critic error, resulting in 24 anomaly regions for Test 1 for both temperature and pressure signals and 21 and 26 for temperature and pressure in Test 2, respectively.

Regarding the hypothesis, based on the analysis of boxplots for Test 1 and Test 2 (see Figure 16 in Chapter 4.4), that the number of anomalies in temperature signals should surpass the number of anomalies in pressure signals: for Test 1, 4 out of 6 error combinations support the hypothesis. For Test 2, 5 out of 6 error combinations indicate that the number of anomalies in the temperature signal is smaller than in the pressure signal.

The second hypothesis, referring to the third research question, aimed to determine whether the number of anomalies in temperature and pressure signals is higher in Test 2 signals than in Test 1 signals. Only one combination, namely the multiplica-



(a) Test for error plots and reconstructed signal for critic + DTW.



(b) Test for error plots and reconstructed signal for critic × DTW

Figure 19: The reconstruction error plots results of applying the TadGAN method to daily average temperature measurements from 01.01.1961 to 31.12.1980. The blue line shows the real signal, the orange line shows the reconstructed signal, and the red line shows the error plot. The abscissa axis refers to timestamps, and the ordinate axis refers to values.

tion of DTW and C_x Critic error, supports the hypothesis. It is also important to note that this combination was mentioned in Geiger et al. (2020) as the one with the best performance for TadGAN.

5.3.3 Comparison of different methods

To compare the performance of the ARIMA and TadGAN methods using the error computation method DTW × critic as a model to test both hypotheses about anomalies in the WD Dataset. As shown in Table 18, TadGAN detected the highest number of anomalies in all four quarters for the pressure signal and in quarters Q2-Q4 for the temperature signal. The highest number of anomalies using the ARIMA method were detected in Q1 for temperature and in Q1 and Q4 for the pressure signal. TadGAN detected the most anomalies in Test 1 in Q1, but the number of detected anomalies in Test 2 was comparable for each quarter for both pressure and temperature signals.

It was also observed that most anomalies in the Temperature collection were detected in the cold months (Q1 and Q4) as well as in the warm months (Q2 and Q3). For the Pressure collection, the distribution of anomalies across quarters, as detected by all algorithms collectively, is more uniform.

In the next Chapter 6 the provided results are discussed in detail.

Table 18: The number of anomalies detected by the ARIMA and TadGAN methods in temperature and pressure test signals of the WD Dataset for each of the 4 quarters of the year. The **bold** indicates the recognized anomalies within all test signals in each quarter.

Method, test signal	Temperature				Pressure			
	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4
ARIMA, Test (01.2001 - 12.2020)	**9**	2	2	**2**	**5**	2	3	5
TadGAN, Test 1 (01.1961 - 12.1980)	**9**	3	-	1	1	4	4	5
TadGAN, Test 2 (01.2001 - 12.2020)	5	**4**	**6**	**2**	**5**	**6**	**7**	**7**

6 Discussion

The first research question of this thesis was to assess whether the results presented in articles about TAnoGAN Bashar and Nayak (2020) and TadGAN Geiger et al. (2020) are reproducible. A deliberate decision was made to thoroughly examine the methodology and hyperparameters while ensuring the utilization of the original code provided by the authors. Regarding TAnoGAN, Bashar and Nayak Bashar and Nayak (2020) have provided code in their GitHub repository,[16] for TadGAN, Geiger et al. (2020) reference a GitHub repository named Orion-ML library,[17] which, according to the introduction, can be installed on a computer and subsequently utilised with pre-programmed pipelines and prepared signals from NAB Dataset.

As demonstrated in Chapter 5.2, similar results are reproducible. However, achieving consistent results posed a significant challenge. In both methods, gaps in the ground truth were examined, which were difficult to notice before the reproduction. The first challenge refers to the lack of information about the chosen thresholds for each dataset, that the authors of the TAnoGAN method Bashar and Nayak (2020) were chosen to test this method. The threshold = 12.2 mentioned on their GitHub repository[18] was presented for the signal *ambient_temperature_system_failure.csv*. This threshold, however, fails to correctly classify any anomalies in other signals within the NAB dataset, for instance, in the signal *ec2_network_in_257a54*. As Bashar and Nayak (2020) mentioned in their article, thresholds should be chosen visually based on the anomaly score plot. Consequently, it can be concluded that the thresholds chosen for reproduction in this thesis may lead to differences in results.

Furthermore, Bashar and Nayak (2020) did not provide numerical results. Nevertheless, based on the performance plot presented in their paper, the F_1-Score results span the range of $[0.4, 0.9]$, while the results obtained within this study fall within the range of $[0.22, 0.93]$, considering the entire range of thresholds used for reproduc-

[16] https://github.com/mdabashar/TAnoGAN/tree/master
[17] https://github.com/sintel-dev/Orion
[18] https://github.com/mdabashar/TAnoGAN/tree/master

tion. Another reason for the lack of exact results could be the absence of a random seed setting in the code, which could have facilitated result reproducibility.

However, it is important to note that TAnoGAN exhibits better performance on collections with short signals compared to collections with long signals. This confirms the hypothesis proposed by Bashar and Nayak (2020), which suggests that TAnoGAN performs better on datasets with a smaller number of measurements (up to 15,000 per signal or time series) than on datasets with longer signals containing a larger number of measurements.

The challenges encountered during the reproduction of TadGAN were as follows. The authors of the TadGAN method Geiger et al. (2020) did not provide guidance on some essential hyperparameters in their paper. The authors of the Orion-ML library Alnegheimish et al. (2022) intended it to be used as a standard Python library, for which specific pipelines were programmed to reproduce TadGAN's results. However, during the research process, attempts to install this library with various recommended configurations were unsuccessful.[19]

The TadGAN method was implemented using functions and hyperparameters for TadGAN pipelines from the Orion-ML library,[20] some of which may potentially differ from the parameters used in Geiger et al. (2020), such as learning rate and epochs.

The second gap was identified in the training process. Geiger et al. (2020) wrote that the model was trained on a single signal from the dataset. However, there is no information regarding whether this was one signal per collection or one signal for the entire NAB dataset. Additionally, there is no information about the name of the test signal. In this thesis, the TadGAN method was applied independently to each signal from every collection within the NAB dataset.[21]

Furthermore, the differences in performance can also be attributed to the fact that Geiger et al. (2020) did not specify how their results, as represented in the article, were obtained. It remains unclear whether these values were derived from medians, modes, or the application of TadGAN to specific signals within the collections. In this study, the best result for each individual collection, surpassing the result indicated in Geiger et al. (2020), as well as the mean result for each collection in the NAB dataset, were used as the outcomes.

Additionally, it was observed that the TadGAN method is more effective in anomaly detection than TAnoGAN, based on the analysis of performance metrics such as accuracy, precision, recall, and the F_1-Score.

Within the scope of the second research question, a hypothesis was formulated that the parameters of the signals themselves should influence the performance of state-of-the-art methods. Bashar and Nayak (2020) postulated that signal length

[19] An error occurred during the configuration of the third-party library ml-stars from sintel `https://github.com/sintel-dev/ml-stars`

[20] `https://github.com/sintel-dev/Orion`

[21] the same procedure as demonstrated in tutorial from Orion -ml library `https://github.com/sintel-dev/Orion/blob/master/tutorials/tulog/Tulog.ipynb`

impacts the performance of TAnoGAN. In this study, signal stationarity was selected as a parameter because stationarity plays a significant role in anomaly detection using statistical statistical methods. Two different but standard tests were chosen to assess stationarity: the ADF and KPSS tests. As shown, choosing different parameters for the test can lead to different results in signals classification between TS-Stationary, Stationary of DS-Stationary. During this study, only the parameter for linear regression was varied, while the maximum number of lags or an information criterion for ADF was set at *default*. It was demonstrated that the performance of both selected state-of-the-art methods depends on the type of stationarity in time series data.

The assumption was that, in general, signals classified as stationary should yield better results when applying the methods to them compared to DS-Stationary signals. Moreover, the results of applying the methods to stationary signals should outperform the results on TS-Stationary signals. When examining the average values across all signal collections in the NAB dataset, it was noticed that for all regression parameters, the combinations of statistical tests yielded superior results for DS-Stationary signals in TadGAN compared to Stationary and TS-Stationary signals. The TAnoGAN algorithm aligns with the assumption, as its results for all regression parameters and combinations of statistical tests are lower for DS-Stationary signals compared to Stationary and TS-Stationary signals.

This may suggest that the effectiveness of TadGAN on non-stationary time series makes its real-world applications quite efficient since time series data in real-world scenarios often exhibit non-stationary behaviour more frequently than true stationarity. However, it's worth noting that the number of collections with DS-Stationary signals is smaller than those with TS-Stationary and Stationary signals. This might potentially lead to errors in calculating the average. Further experiments on datasets containing more DS-Stationary signals are necessary to continue this investigation.

In the third research question, the intention was to apply the ARIMA and TadGAN algorithms to real weather data from Bamberg. For this thesis, temperature and pressure data recorded every hour (temperature) and every three hours (pressure) over a span of 70 years were selected. The average daily temperature was calculated to reduce the data volume. Then the entire signal was divided into three independent segments: a training set from 1980 to 2000, test 1 from 1961 to 1980, and test 2 from 2001 to 2020.

The main hypothesis was that the number of anomalies in test 2 should be greater than the number of anomalies in the first test. However, applying TadGAN with the parameters from the experiments, provided for research question 1, yielded results indicating a maximum of seven anomalies within the test signals, and each anomaly region has a length of 120 days. The length of the anomaly measurement appears somewhat unrealistic. Therefore, the parameters for the anomaly detection window were adjusted. As a result, the detected anomaly regions had a length of 36 days. While these window-size adjustments were suitable for the experiments in this paper, it may be worthwhile for future research to explore different window-size parameters, especially if it leads to the reduction of the anomaly region.

Moreover, after applying the TadGAN method with different parameters, it was observed that only one parameter setup ($DTW \times critic$ supported both hypotheses regarding the WD dataset: the predominance of anomalies in pressure data and an increase in the number of anomalies for both temperature and pressure in Test 2, which includes observations from 2001 to 2020. It is also important to note that this parameter demonstrated the best mean performance in all datasets under study in the TadGAN paper Geiger et al. (2020).

It is important to emphasise that ARIMABox et al. (2015) relies on historical data to make predictions. This means that predicting several steps ahead, especially with the chosen signal from WD dataset, was a significant challenge because of the length of the data and the period between observations (one day for TS-Stationary time series with an additional seasonal component for a temperature signal). Furthermore, attempts to fit an ARIMA model with varying parameters resulted in AIC criteria reaching up to seven digits or even becoming infinite. This indicates that the model is suboptimal, and its forecasted values are far from real values in the test signal. To address this issue, a decision was made to preprocess the data again for the ARIMA model. This involved calculating the average monthly temperature, which allowed for the selection of appropriate parameters to fit the ARIMA model.

The anomalies identified by both models can be compared because point anomalies were detected with the ARIMA model (months with anomalous temperature and pressure measurements). On the other hand, parameters for TadGAN were so selected that the anomaly window had a length of 36 days.

However, it's essential to note that ARIMA and TadGAN are sensitive to the choice of hyperparameters. The primary approach in detecting anomalies in these methods is to estimate the reconstruction error. Depending on how this estimation is carried out and the threshold chosen, it is possible to obtain significantly different results for the same signal. The experiments with the WD dataset using the TadGAN method clearly illustrated that varying the parameter choices can yield vastly different outcomes for the same signal.

7 Conclusion

This study was focused on the following research questions:

1. Is it possible to replicate the results of two selected state-of-the-art GAN-based methods for anomaly detection? What are the key differences in terms of performance between these methods, and how is the performance of chosen GAN methods compared with the classical statistical ARIMA method?

2. The time series data is analysed with regard to different types of stationarity using classical statistical tests: ADF and KPSS. How do the different types of stationarity influence the performance of selected GAN-based methods?

3. The best selected state-of-the-art GAN-based method is used to detect anomalies in an unlabelled Weather dataset for temperature and pressure observations in Bamberg for 1961-2020. For comparison, a classical statistical ARIMA method is applied. How does the amount of detected anomalies differ depending on the selected method? Is the number of detected anomalies similar for temperature and pressure modalities in the Weather dataset? Is the number of anomalies in the selected Weather dataset increased for 2000-2020 compared to the observations for 1961-1980?

To answer these questions, the following dataset and state-of-the-art methods were selected

Datasets To replicate the results from Geiger et al. (2020) and Bashar and Nayak (2020) and investigate the dependencies between dataset properties and the obtained results, the labelled NAB dataset Ahmad et al. (2017), which comprises both real and artificial time series signals, was used in this study. An unlabelled WD dataset was created to apply the selected methods on weather data, containing data about the daily average temperature and pressure in Bamberg (weather station 282) from 01.01.1961 to 31.12.2020. Each signal in both datasets was examined for stationarity using two classical statistical tests: ADF Dickey and Fuller (1979) and KPSS Kwiatkowski et al. (1992).

Methods For the detection of stationarity in signals within the NAB and WD datasets, the following parameters for statistical tests were selected:

ADF Test: maxlag = None, regression = {c, ct , ctt , n}, autolag = AIC
KPSS Test: nlags = auto, regression = {c,ct}

To reproduce the results of the selected state-of-the-art GAN-based methods, TAnoGAN Bashar and Nayak (2020), which utilizes a classic GAN approach, and TadGAN Geiger et al. (2020), an enhanced novel GAN-based method, were chosen. As a baseline method, a classical statistical ARIMA Box et al. (2015) method was selected.

TAnoGAN has the following parameters: The generator consists of a 3-layer LSTM with 32, 64, and 128 hidden units in each layer. The discriminator is a 1-layer LSTM with 100 hidden units. Window size and window step were $s_w = 60$, $s_t = 1$ for training, $s_t = 30$ for anomaly detection. As the optimizer, Adam with a learning rate of 0.0002 was used.

TadGAN has the following parameters: Generator $\mathcal{E}$ is a single-layer bidirectional LSTM with 100 hidden units. Generator $\mathcal{G}$ is a two-layer bidirectional LSTM network with 64 hidden units in each layer. The discriminators C_x and C_z are two 1D CNN. Window size and window step were $s_w = 100$, $s_t = 1$. As the optimizer, Adam with a learning rate of 0.0005 was used.

For anomaly detection in the WD Dataset, the following ARIMA models were chosen: SARIMA(0,0,0)(1,0,2)[12] for temperature signals and SARIMA(0,0,2)(0,0,1)[12] for pressure signals.

Evaluation results The performance of the labelled NAB dataset was evaluated using standard evaluation metrics, including accuracy, precision, recall, and the F_1-Score. However, for the WD Dataset, the performance was not measured in terms of metrics; rather, only the number of detected anomalies was considered.

Within the study, experiments were conducted to obtain the following results, which address three research questions:

Reproduction results During the study, it was observed that the results obtained using the selected methods were only partially reproducible. This discrepancy was attributed to the absence of certain default hyperparameters used by the authors of the original research Bashar and Nayak (2020), Geiger et al. (2020), which can significantly impact the final results. Additionally, there was uncertainty in obtaining the final values of the F_1-Score within the NAB dataset signals, as indicated in the original study Bashar and Nayak (2020), Geiger et al. (2020).

Furthermore, it was noted that the TadGAN method outperformed the TAnoGAN method in terms of accuracy, precision, and the F_1-Score. This observation suggests that this performance advantage led to selecting the TadGAN method as the approach to address research question 3.

Influence of signals stationarity The stationarity analysis of signals from the NAB dataset has shown that the results of the stationarity analysis depend on the chosen methodology for testing stationarity in time series. This means that different types of stationarity can be detected within one signal based on the selected hyperparameters of statistical tests. Through experimental setups in this study, it was discovered that the performance of the selected state-of-the-art methods is dependent on the type of stationarity exhibited by the signals. However, the DS-Stationary signals exhibit a trade-off between precision and recall for both selected methods, with low precision and high recall.

Anomaly detection in WD dataset Applying the selected GAN method (TadGAN) for anomaly detection in weather data revealed the following results. The TadGAN algorithm detected more anomalies than the ARIMA algorithm, which may suggest a higher capability of GANs in identifying complex patterns in unlabeled data. Additionally, the results of applying TadGAN supported the hypothesis, based on the original data analysis, that temperature data contains fewer anomalies than pressure data.

However, it is important to note that the main hypothesis, which posited an increase in the number of anomalies in weather data over the past 20 years, was confirmed only for one combination of reconstruction errors used to replicate the results with the TadGAN method. This particular combination also yielded the best results in the original study Geiger et al. (2020). It is worth mentioning that the parameters defining local adaptive thresholding for unlabelled data should be chosen empirically, as they can lead to different results.

This study observed that signal parameters, such as stationarity, can influence the performance of selected algorithms. The code for this thesis is available on GitHub.[22]

Future works This work opens several avenues for further research. Firstly, within the scope of this study, it was found that the results are dependent solely on one statistical parameter of time series stationarity. Time series data contain additional statistical parameters, including linearity. Investigating whether this parameter influences the results of the selected algorithms is a potential area for future research. Secondly, NAB and WD signals were examined for stationarity only under different regression parameters. The other parameters were not considered and the further classification of signal properties might potentially have an influence on the results that war in the study becommen.

Additionally, this study primarily focused on two algorithms, while in the GAN family, there exist more options. Future extensions should consider exploring the influence of GAN-based methods. There are also several univariate datasets, including the mentioned Yahoo Webscope, which contains a substantial number of point anomalies unlike the NAB dataset used in this work. Investigating the effectiveness of anomaly detection methods on point anomalies using selected methods also presents an interesting avenue for research.

In this study, the primary focus was on univariate time series analysis. However, the same research question could be extended to explore multivariate time series. Within the GAN family, there exist algorithms designed to handle multivariate time series, such as MAD-GAN. In the context of real-world data, it would be intriguing to analyze these data sets collectively. For instance, creating a multivariate time series from weather data, incorporating variables such as temperature, pressure, as well as additional parameters like illumination and humidity, could not only validate the anomalies identified in this study but also potentially unveil new ones.

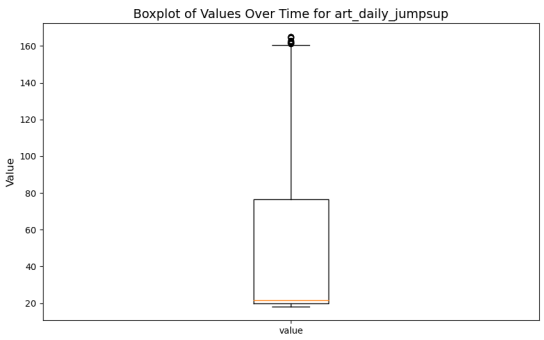
[22] https://github.com/anjahr/masterthesis-anomalie

A Appendix

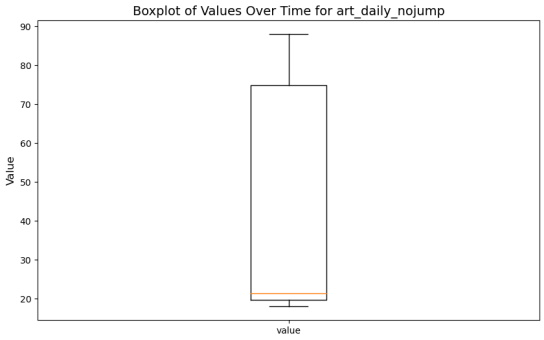
A.1 Boxplots of NAB and WD Datasets

A.1.1 Boxlots of NAB Datset

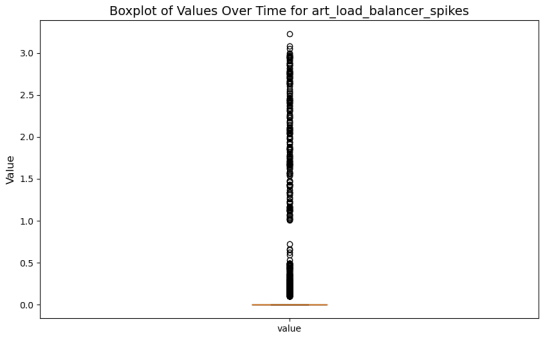
Figure 20 shows boxplots for the ArtificialWithAnomaly collection, Figure 21 shows boxplots for the RealAdExchange collection, Figures 22, 23, 24 show boxplots for the RealAWSCloudwatch collection, Figure 25 shows boxplots for the RealKnownCause collection, Figure 26 shows boxplots for the RealTraffic collection, Figures 27, 28 show boxplots for the RealTwittes collection.



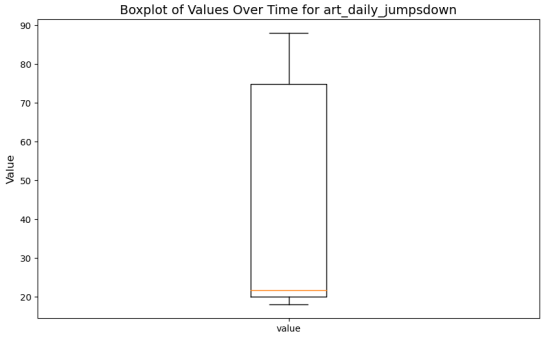
(a) *art_daily_jumpsup* signal



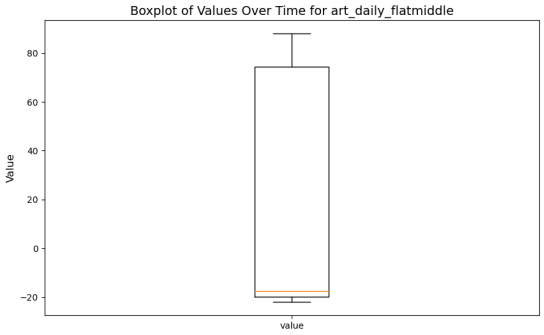
(b) *art_daily_no_jump* signal



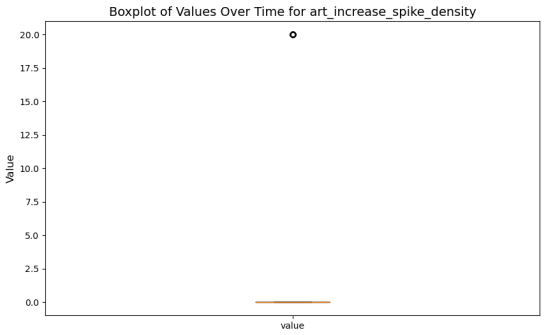
(c) *art_load_balancer.png* signal



(d) *art_daily_jumpsdown* signal

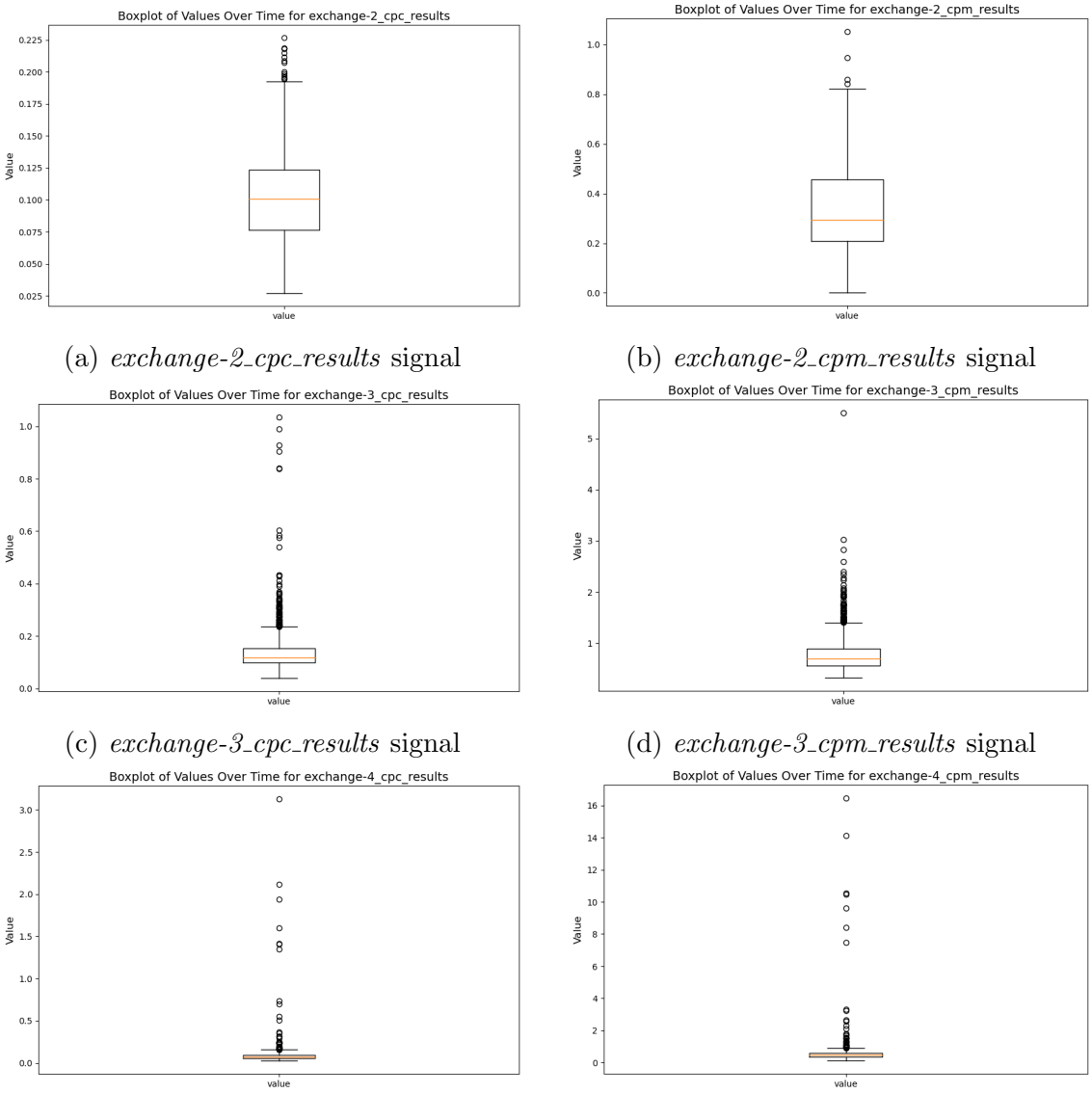


(e) *art_daily_flatmiddle* signal



(f) *art_increase_spike_density* signal

Figure 20: Boxplots for an Artificial with Anomaly collection. The horizontal orange line represents the median, the upper limit of the black box indicates Q3, and the lower limit shows Q1. The upper and lower whiskers point to borders for normal data. Points refer to potential anomalies.



(a) *exchange-2_cpc_results* signal

(b) *exchange-2_cpm_results* signal

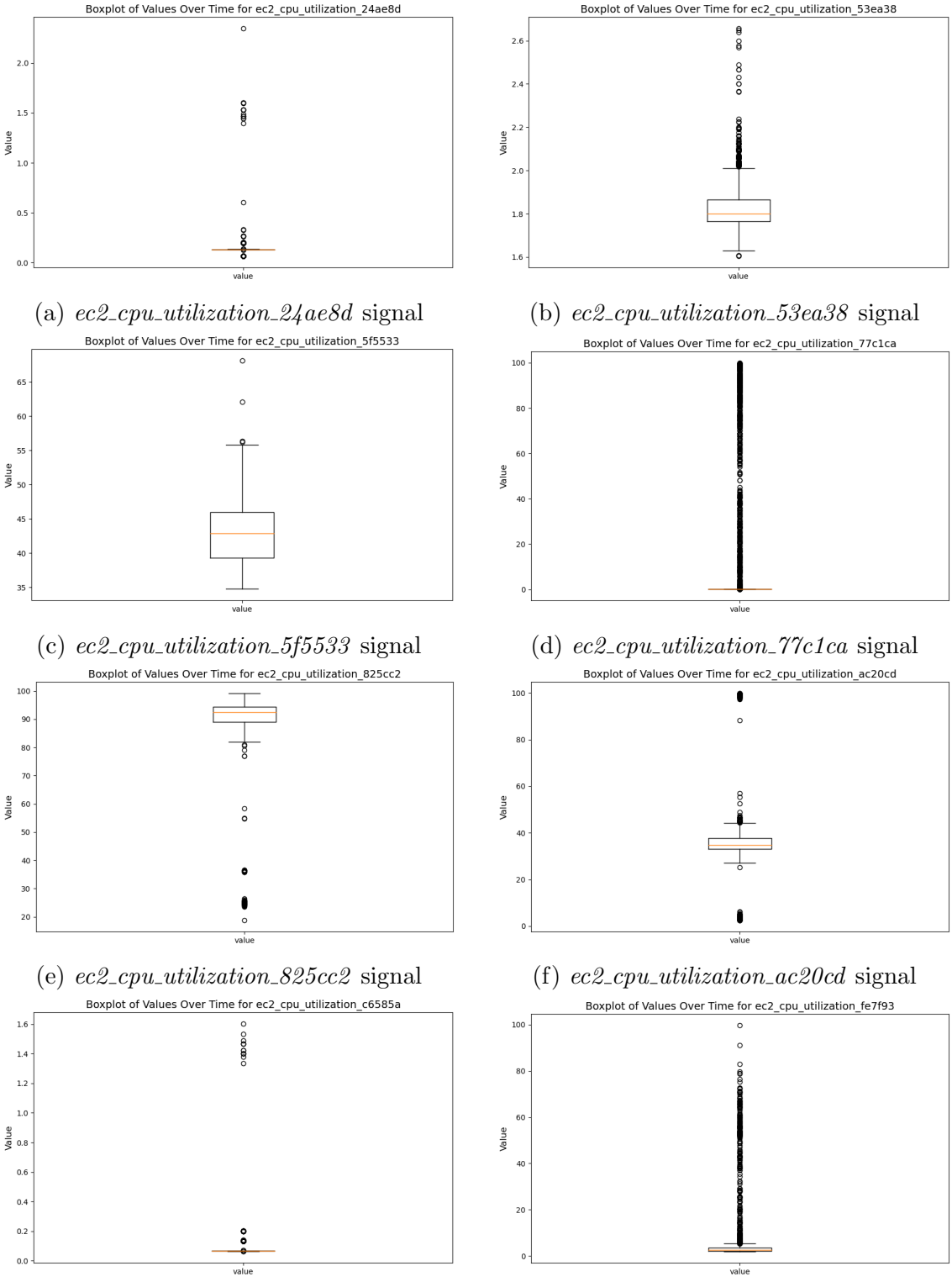
(c) *exchange-3_cpc_results* signal

(d) *exchange-3_cpm_results* signal

(e) *exchange-4_cpc_results* signal

(f) *exchange-4_cpm_results* signal

Figure 21: Boxplots for an AdExchange collection. The horizontal orange line represents the median, the upper limit of the black box indicates Q3, and the lower limit shows Q1. The upper and lower whiskers point to borders for normal data. Points refer to potential anomalies.



(a) *ec2_cpu_utilization_24ae8d* signal

(b) *ec2_cpu_utilization_53ea38* signal

(c) *ec2_cpu_utilization_5f5533* signal

(d) *ec2_cpu_utilization_77c1ca* signal

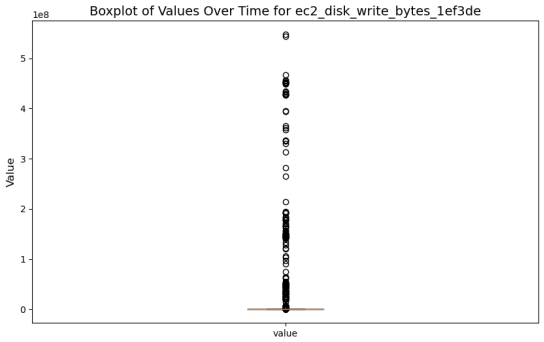
(e) *ec2_cpu_utilization_825cc2* signal

(f) *ec2_cpu_utilization_ac20cd* signal

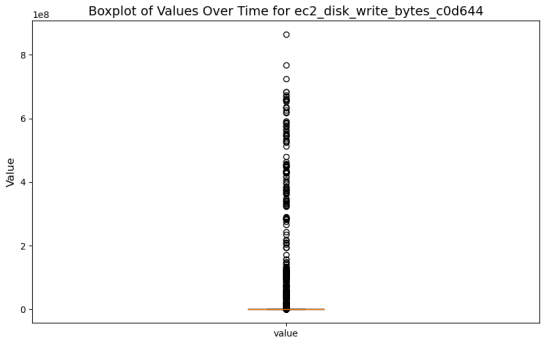
(g) *ec2_cpu_utilization_c6585a* signal

(h) *ec2_cpu_utilization_fe7f93* signal

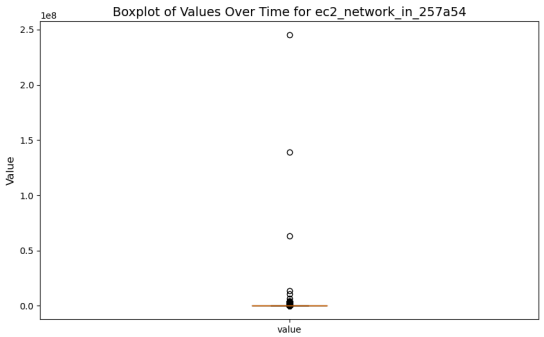
Figure 22: Boxplots for an AWS collection, part 1. The horizontal orange line represents the median, the upper limit of the black box indicates Q3, and the lower limit shows Q1. The upper and lower whiskers point to borders for normal data. Points refer to potential anomalies.



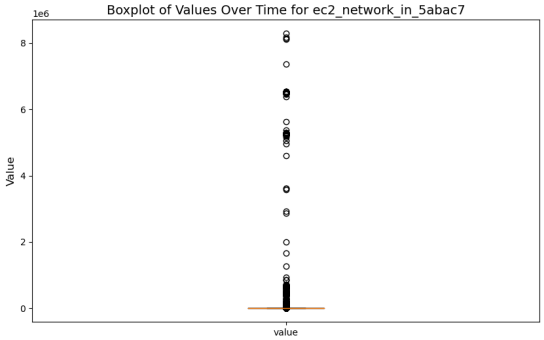
(a) *ec2_disk_write_bytes_1ef3de* signal



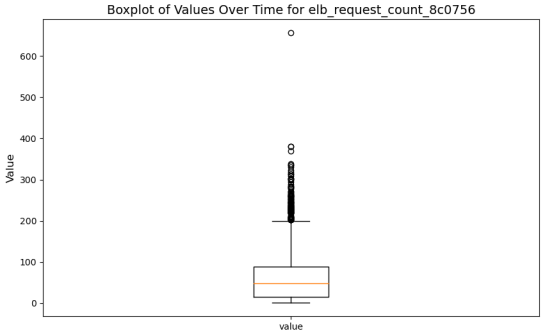
(b) *ec2_disk_write_bytes_c0d644* signal



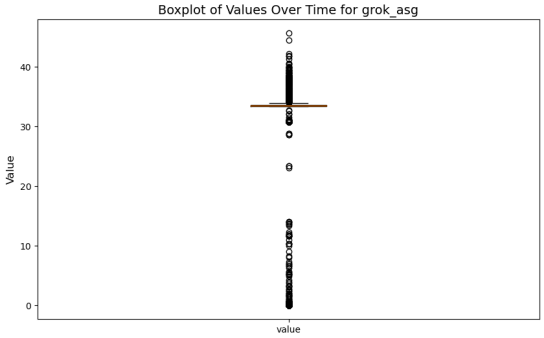
(c) *ec2_network_in_257a54* signal



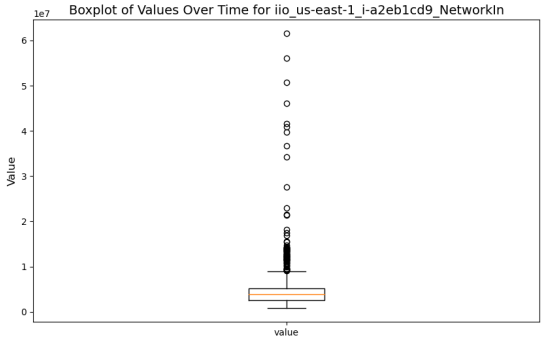
(d) *ec2_network_in_5abac7* signal



(e) *elb_request_count_8c0756* signal

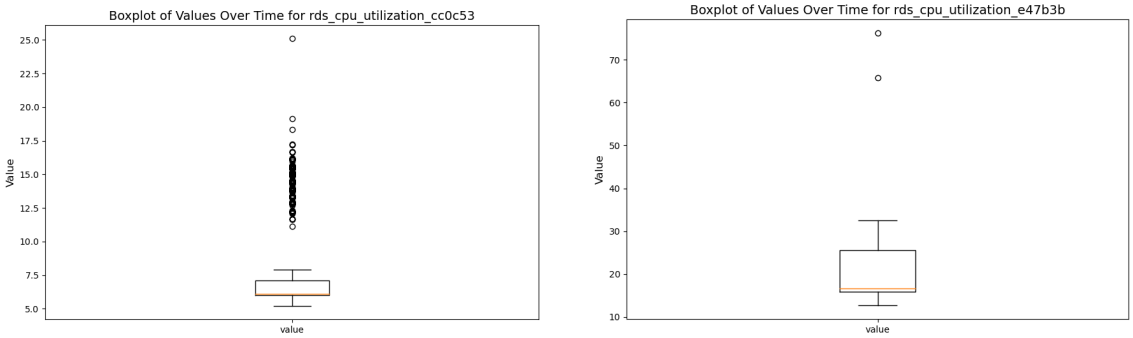


(f) *grok_asg_anomaly* signal



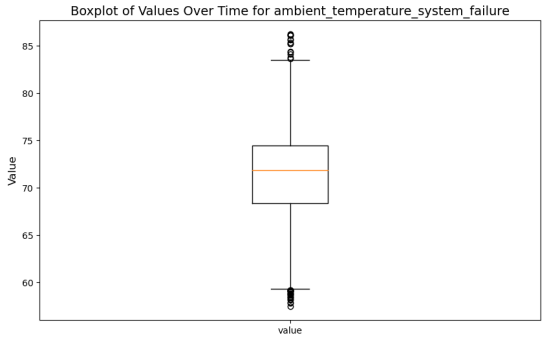
(g) *iio_us-east-1_i-a2eb1cd9_NetworkIn* signal

Figure 23: Boxplots for an AWS collection, part 2. The horizontal orange line represents the median, the upper limit of the black box indicates Q3, and the lower limit shows Q1. The upper and lower whiskers point to borders for normal data. Points refer to potential anomalies.

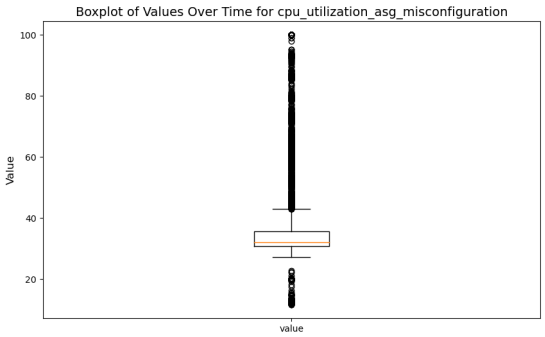


(a) *rds_cpu_utilization_cc0c53* signal (b) *rds_cpu_utilization_e47b3b* signal

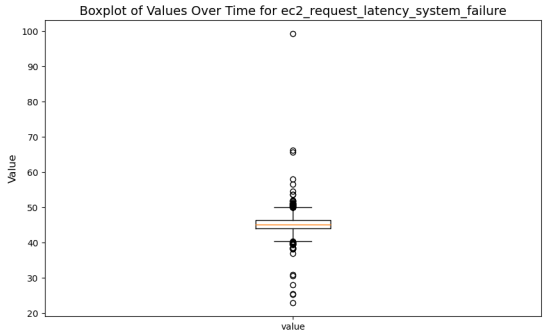
Figure 24: Boxplots for an AWS collection, part 3. The horizontal orange line represents the median, the upper limit of the black box indicates Q3, and the lower limit shows Q1. The upper and lower whiskers point to borders for normal data. Points refer to potential anomalies.



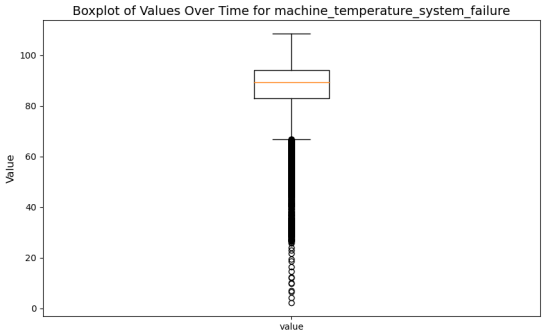
(a) *ambitient_temperature_system_failure* signal



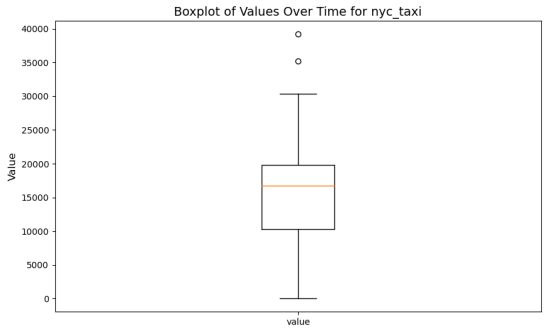
(b) *cpu_utilization_asg_misconfiguration* signal



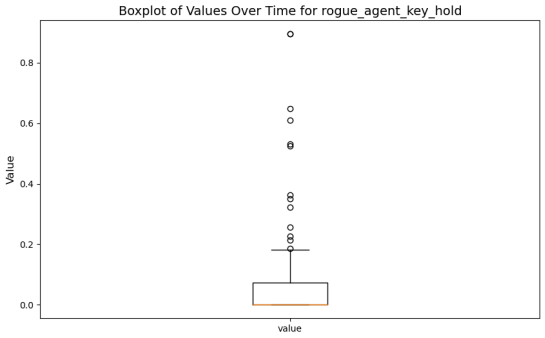
(c) *ec2_request_latency_system_failure* signal



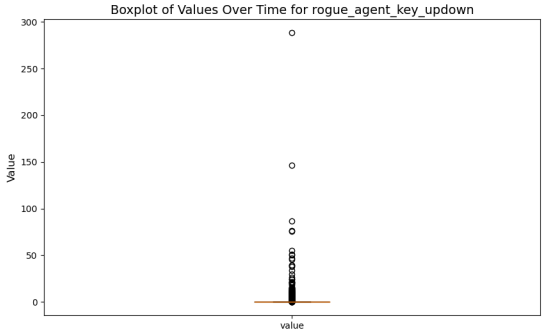
(d) *machine_temperature_system_failure* signal



(e) *nyc_taxi* signal

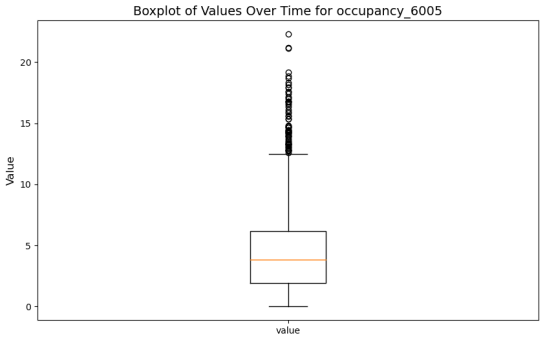


(f) *rogue_agent_key_hold* signal

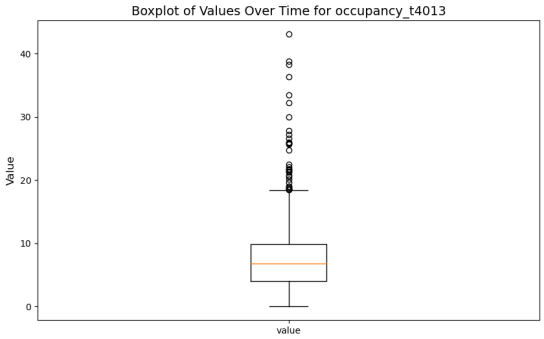


(g) *rogue_agent_key_updown* signal

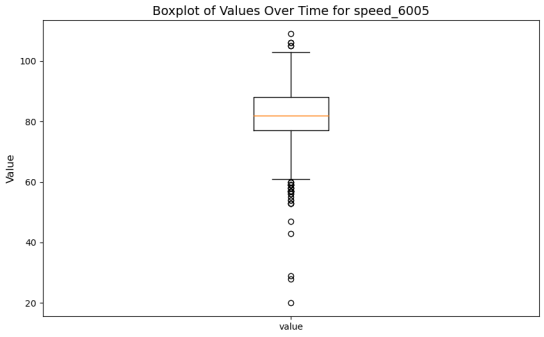
Figure 25: Boxplots for a KnownCause collection. The horizontal orange line represents the median, the upper limit of the black box indicates Q3, and the lower limit shows Q1. The upper and lower whiskers point to borders for normal data. Points refer to potential anomalies.



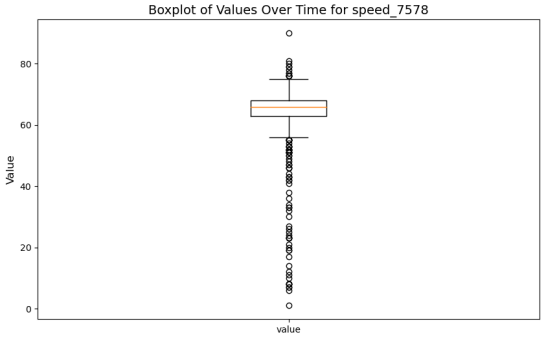
(a) *occupancy_6005* signal



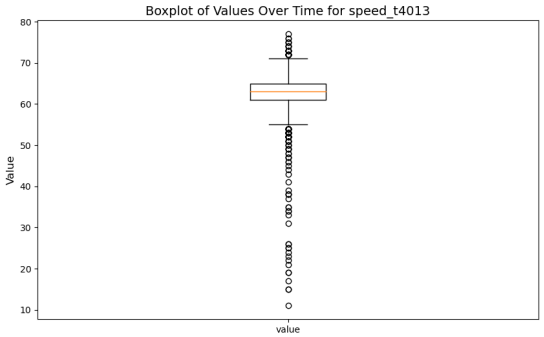
(b) *occupancy_t4013* signal



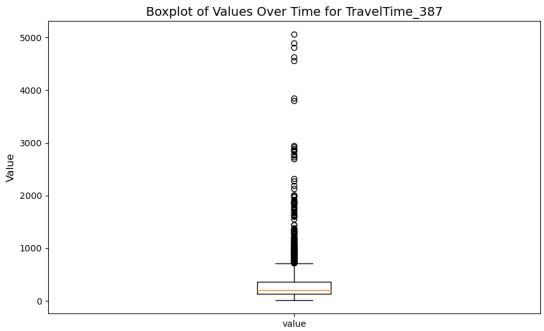
(c) *speed_6005* signal



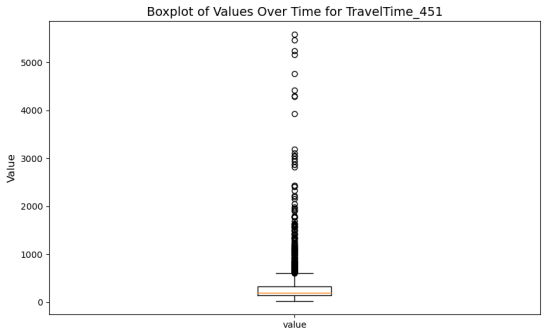
(d) *speed_7578* signal



(e) *speed_t4013* signal

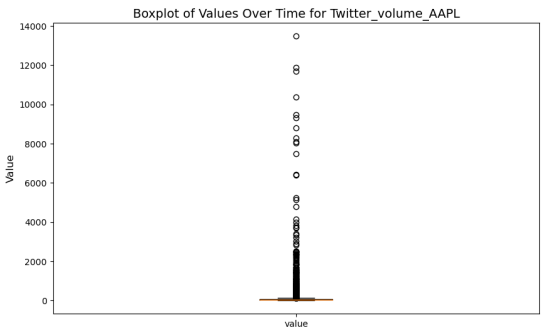


(f) *TravelTime_387* signal

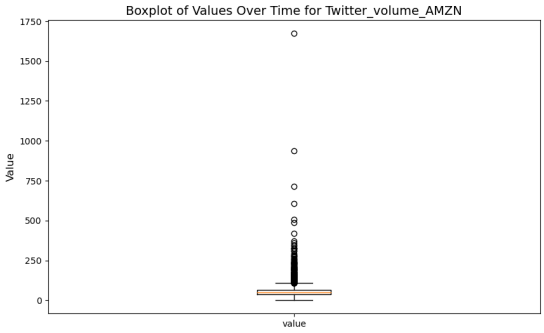


(g) *TravelTime_451* signal

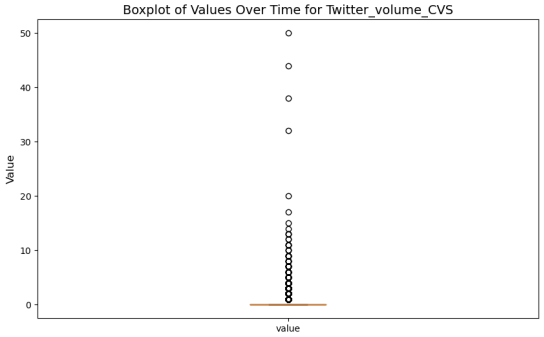
Figure 26: Boxplots for a Traffic collection. The horizontal orange line represents the median, the upper limit of the black box indicates Q3, and the lower limit shows Q1. The upper and lower whiskers point to borders for normal data. Points refer to potential anomalies.



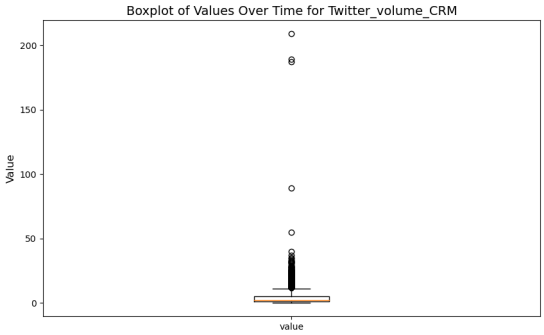
(a) *Twitter_volume_AAPL* signal



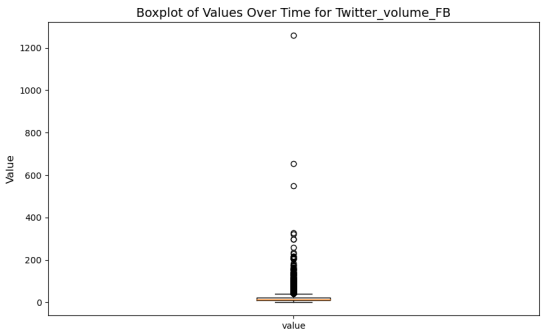
(b) *Twitter_volume_AMZN* signal



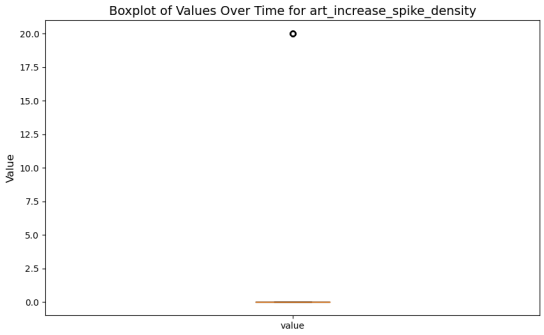
(c) *Twitter_volume_CVS* signal



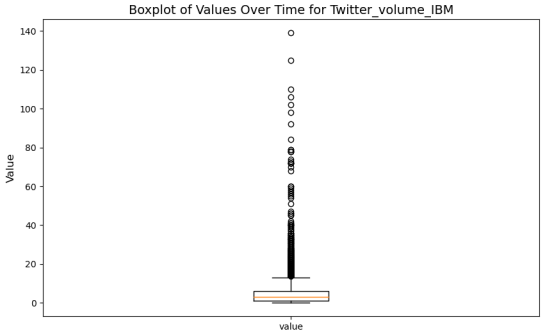
(d) *Twitter_volume_CRM* signal



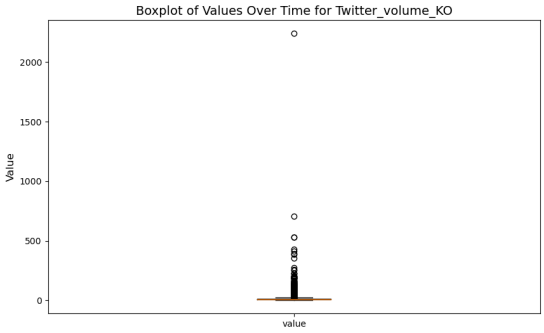
(e) *Twitter_volume_FB* signal



(f) *Twitter_volume_GOOG* signal

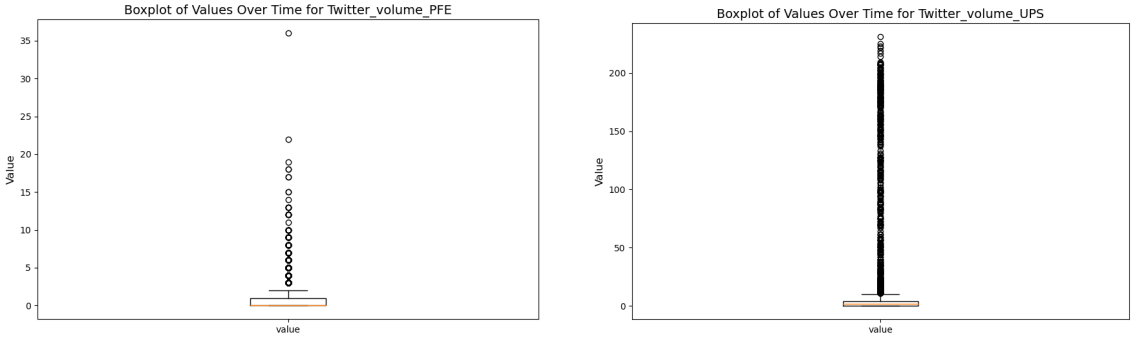


(g) *Twitter_volume_IBM* signal



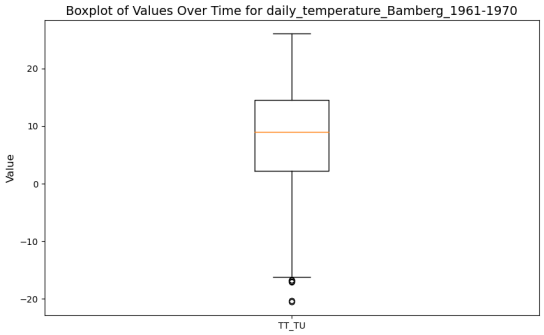
(h) *Twitter_volume_KO* signal

Figure 27: Boxplots for a Twitter collection, part 1. The horizontal orange line represents the median, the upper limit of the black box indicates Q3, and the lower limit shows Q1. The upper and lower whiskers point to borders for normal data. Points refer to potential anomalies.

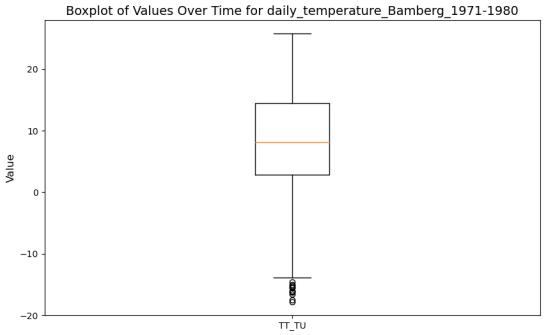


(a) *Twitter_volume_PFE* signal (b) *Twitter_volume_UPS* signal

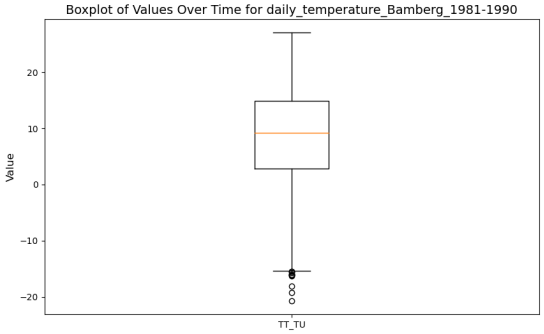
Figure 28: Boxplots for a Twitter collection, part 2. The horizontal orange line represents the median, the upper limit of the black box indicates Q3, and the lower limit shows Q1. The upper and lower whiskers point to borders for normal data. Points refer to potential anomalies.



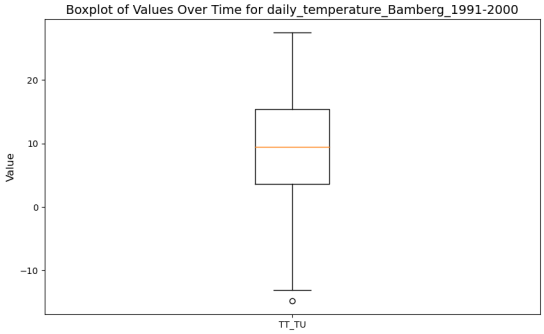
(a) Daily average temperature for 01.01.1961 to 31.12.1970



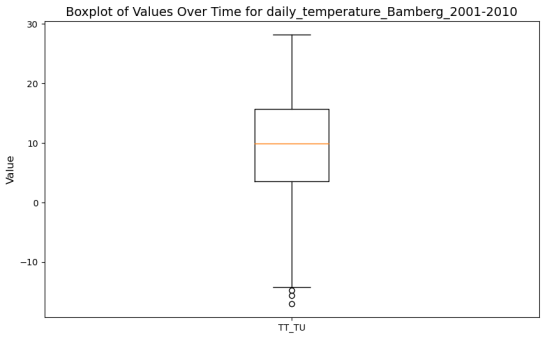
(b) Daily average temperature for 01.01.1971 to 31.12.1980



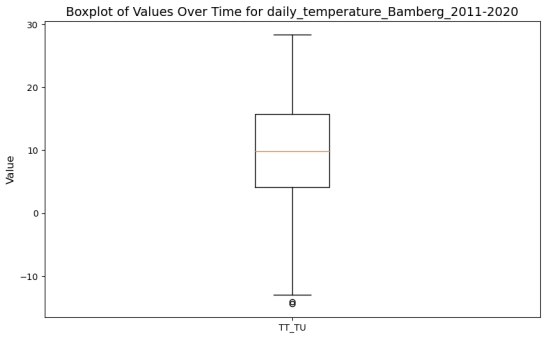
(c) Daily average temperature for 01.01.1981 to 31.12.1990



(d) Daily average temperature for 01.01.1991 to 31.12.2000



(e) Daily average temperature for 01.01.2001 to 31.12.2010

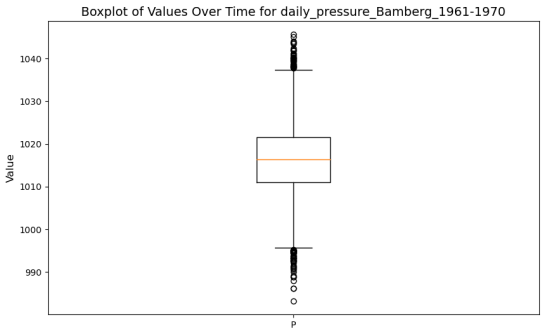


(f) Daily average temperature for 01.01.2011 to 31.12.2020

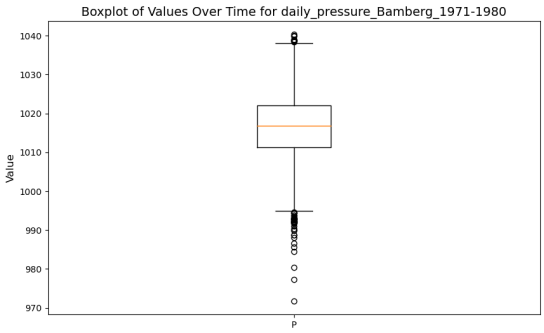
Figure 29: Boxplots for Temperature collection of WD dataset. The horizontal orange line represents the median, the upper limit of the black box indicates Q3, and the lower limit shows Q1. The upper and lower whiskers point to borders for normal data. Points refer to potential anomalies.

A.1.2 WD Dataset Boxplots

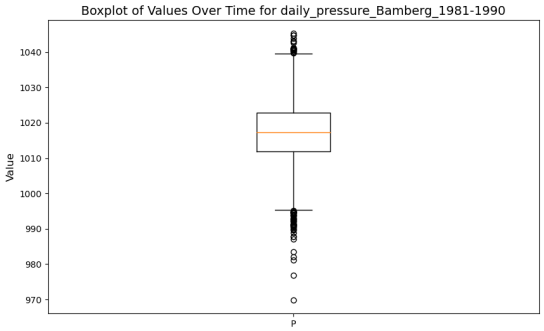
Figure 29 displays boxplots for decade signals of the Temperature collection, while Figure 30 illustrates boxplots for decade signals of the Pressure collection.



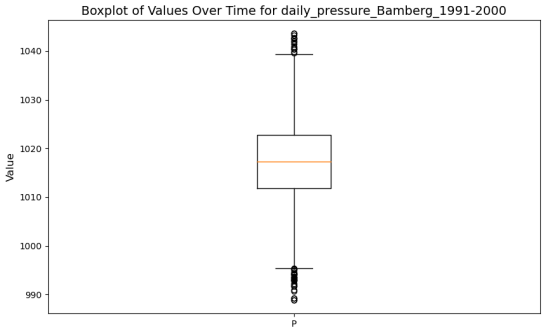
(a) Daily average pressure for 01.01.1961 to 31.12.1970



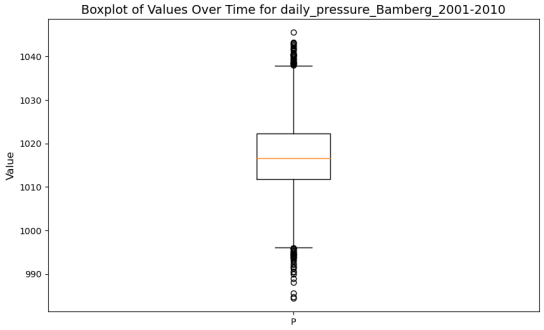
(b) Daily average pressure for 01.01.1971 to 31.12.1980



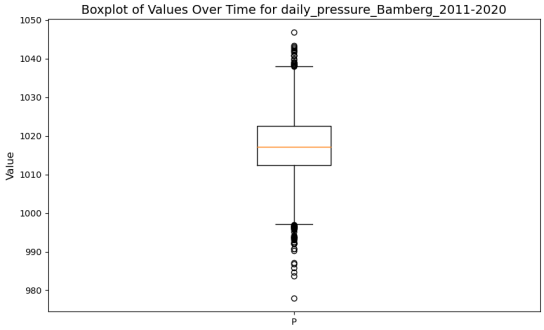
(c) Daily average pressure for 01.01.1981 to 31.12.1990



(d) Daily average pressure for 01.01.1991 to 31.12.2000

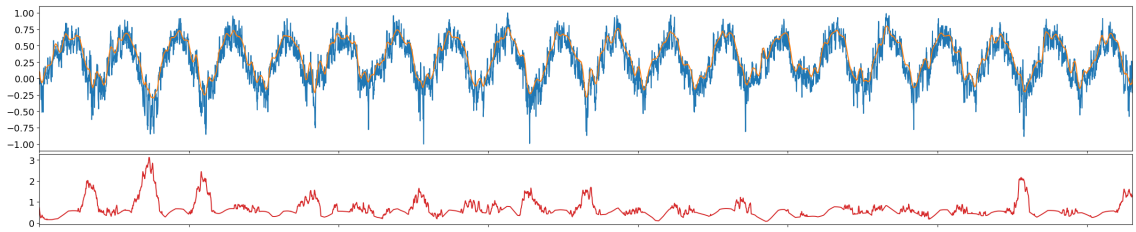


(e) Daily average pressure for 01.01.2001 to 31.12.2010

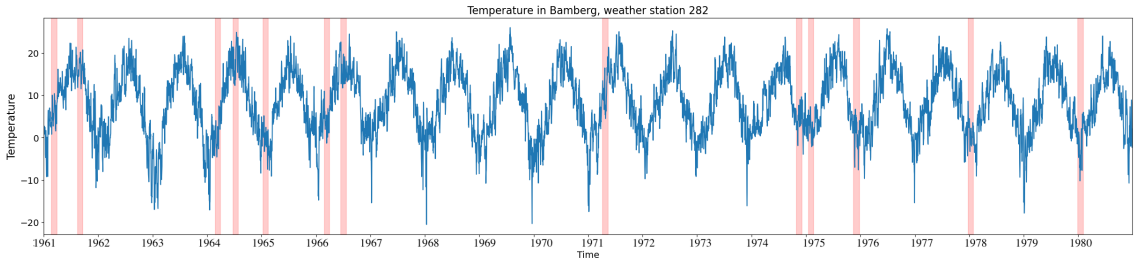


(f) Daily average pressure for 01.01.2011 to 31.12.2020

Figure 30: Boxplots for Pressure collection of WD dataset. The horizontal orange line represents the median, the upper limit of the black box indicates Q3, and the lower limit shows Q1. The upper and lower whiskers point to borders for normal data. Points refer to potential anomalies.



(a) Test 1 error plots and reconstructed signal. The blue line shows the real signal, the orange line shows the reconstructed signal, and the red line shows the error plot.



(b) Test 1 anomalies plot. The blue line represents the real signal, while the red regions indicate anomaly regions.

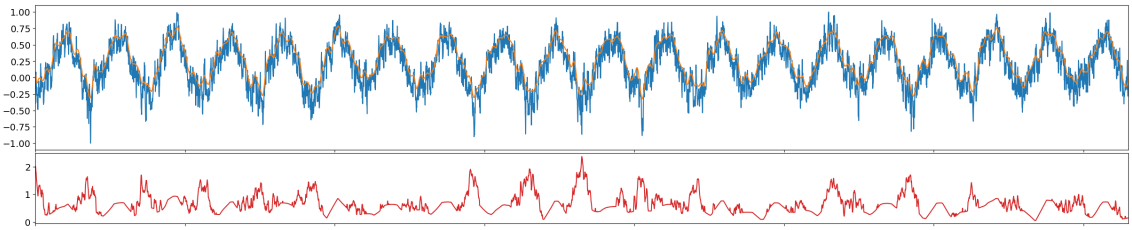
Figure 31: The results of applying the TadGAN method to daily average temperature measurements from 01.01.1961 to 31.12.1980. The reconstructed error was a convex combination of point-wise difference and C_x critic error.

A.2 Error Plots and Detected Anomaly Plots for TagGAN Application on WD dataset

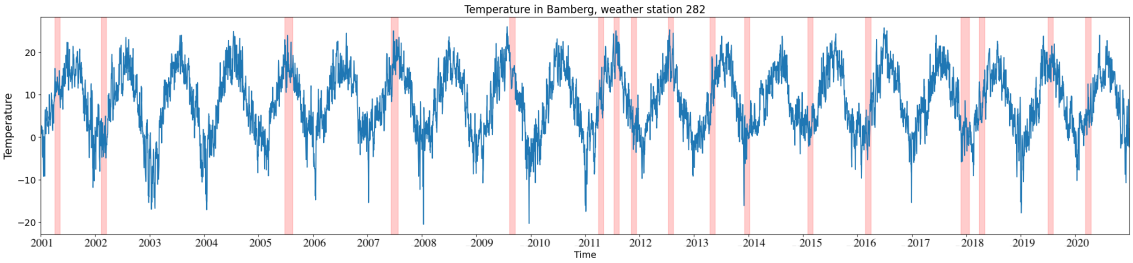
Table 19 displays the combination of possible error types for the anomaly score in TagGAN and the corresponding plots.

Table 19: Assignment of anomaly scores to computational methods of the TadGAN method and plots.

Variation	Figures
critic + point	Figure 31, 32, 33, 34
critic × point	Figure 35, 36, 37, 38
critic + area	Figure 39, 40, 41, 42
critic × area	Figure 43, 44, 45, 46
critic + DTW	Figure 47, 48, 49, 50
critic × DTW	Figure 51, 52, 53, 54

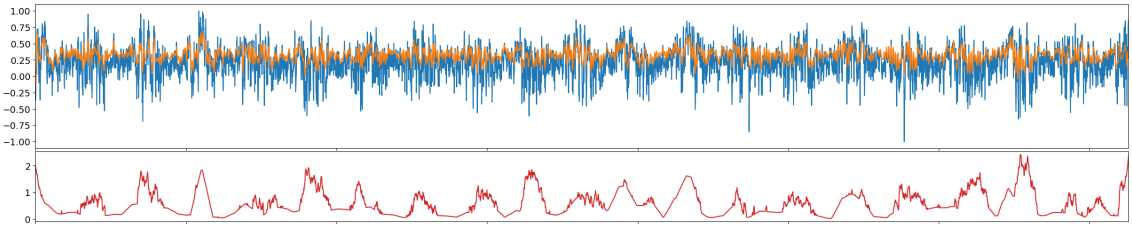


(a) Test 2 error plots and reconstructed signal. The blue line shows the real signal, the orange line shows the reconstructed signal, and the red line shows the error plot.

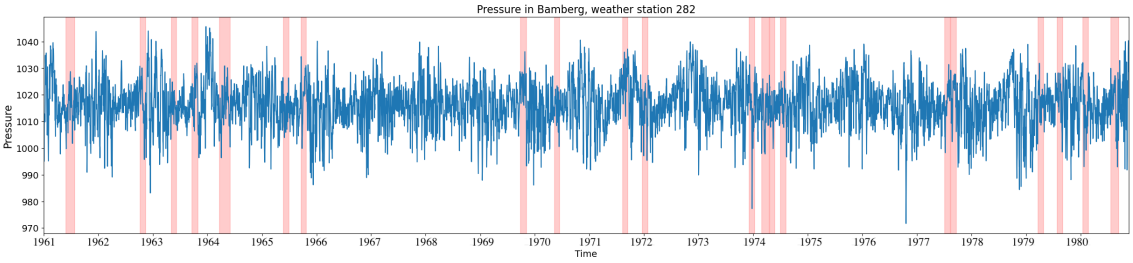


(b) Test 2 anomalies plot. The blue line represents the real signal, while the red regions indicate anomaly regions.

Figure 32: The results of applying the TadGAN method to daily average temperature measurements from 01.01.2001 to 31.12.2020. The reconstructed error was a convex combination of point-wise difference and C_x critic error.

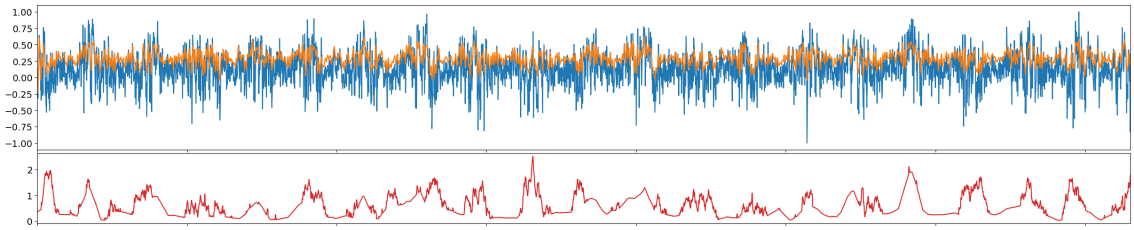


(a) Test 1 error plots and reconstructed signal. The blue line shows the real signal, the orange line shows the reconstructed signal, and the red line shows the error plot.

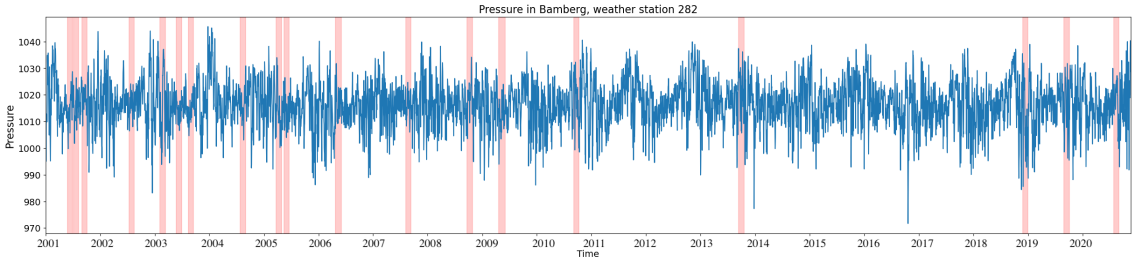


(b) Test 1 anomalies plot. The blue line represents the real signal, while the red regions indicate anomaly regions.

Figure 33: The results of applying the TadGAN method to daily average pressure measurements from 01.01.1961 to 31.12.1980. The reconstructed error was a convex combination of point-wise difference and C_x critic error.

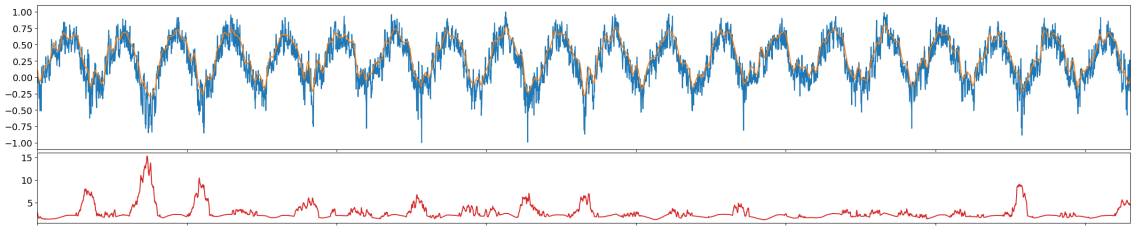


(a) Test 2 error plots and reconstructed signal. The blue line shows the real signal, the orange line shows the reconstructed signal, and the red line shows the error plot.

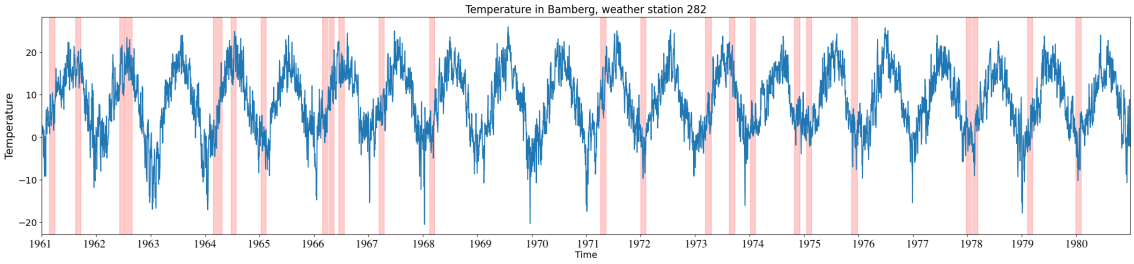


(b) Test 2 anomalies plot. The blue line represents the real signal, while the red regions indicate anomaly regions.

Figure 34: The results of applying the TadGAN method to daily average pressure measurements from 01.01.2001 to 31.12.2020. The reconstructed error was a convex combination of point-wise difference and C_x critic error.

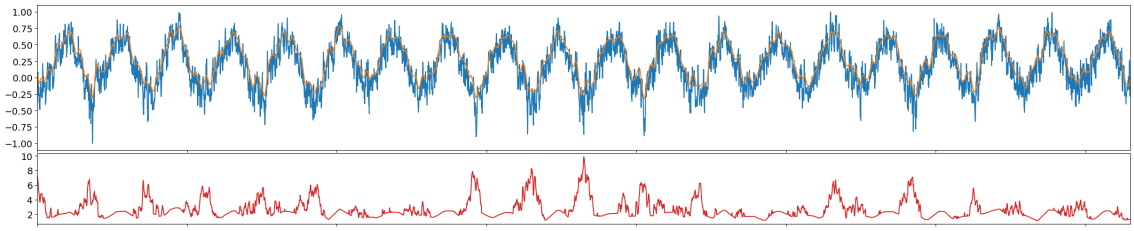


(a) Test 1 error plots and reconstructed signal. The blue line shows the real signal, the orange line shows the reconstructed signal, and the red line shows the error plot.

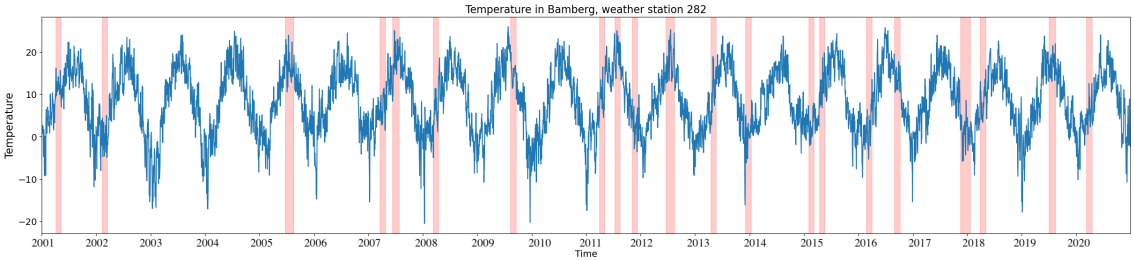


(b) Test 1 anomalies plot. The blue line represents the real signal, while the red regions indicate anomaly regions.

Figure 35: The results of applying the TadGAN method to daily average temperature measurements from 01.01.1961 to 31.12.1980. The reconstructed error was a multiplication of point-wise difference and C_x critic error.

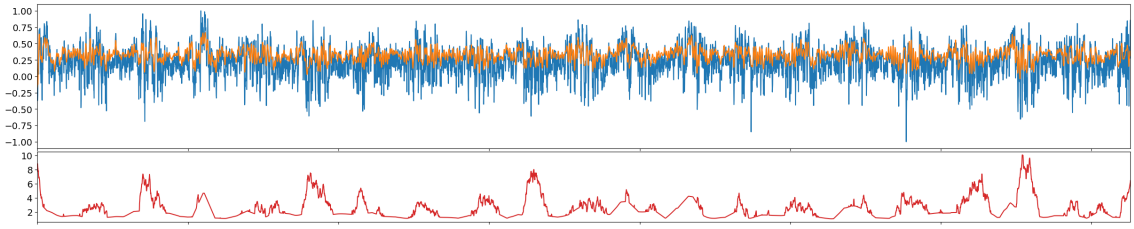


(a) Test 2 error plots and reconstructed signal. The blue line shows the real signal, the orange line shows the reconstructed signal, and the red line shows the error plot.

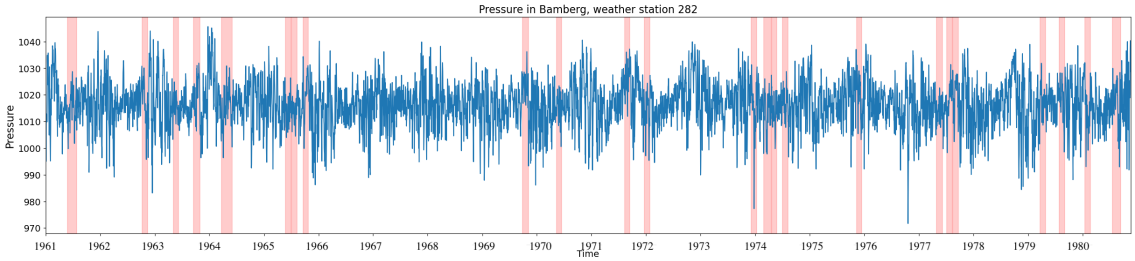


(b) Test 2 anomalies plot. The blue line represents the real signal, while the red regions indicate anomaly regions.

Figure 36: The results of applying the TadGAN method to daily average temperature measurements from 01.01.2001 to 31.12.2020. The reconstructed error was a multiplication of point-wise difference and C_x critic error.

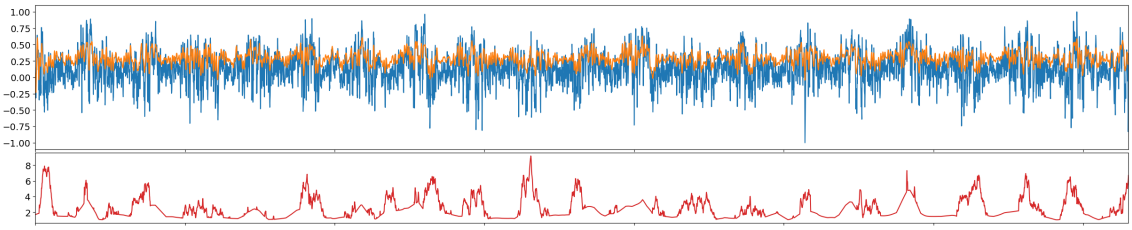


(a) Test 1 error plots and reconstructed signal. The blue line shows the real signal, the orange line shows the reconstructed signal, and the red line shows the error plot.

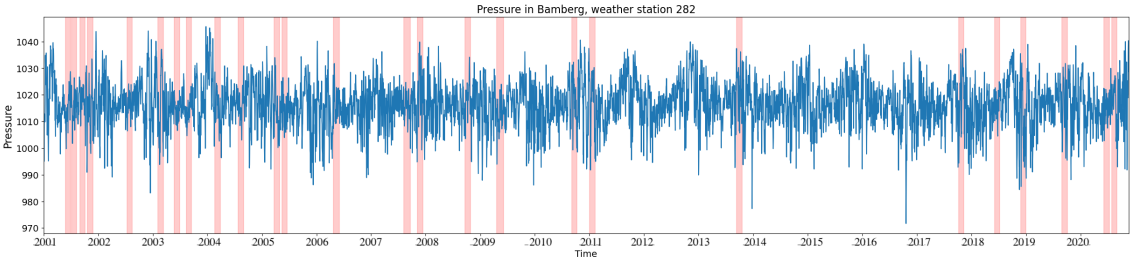


(b) Test 1 anomalies plot. The blue line represents the real signal, while the red regions indicate anomaly regions.

Figure 37: The results of applying the TadGAN method to daily average pressure measurements from 01.01.1961 to 31.12.1980. The reconstructed error was a multiplication of point-wise difference and C_x critic error.

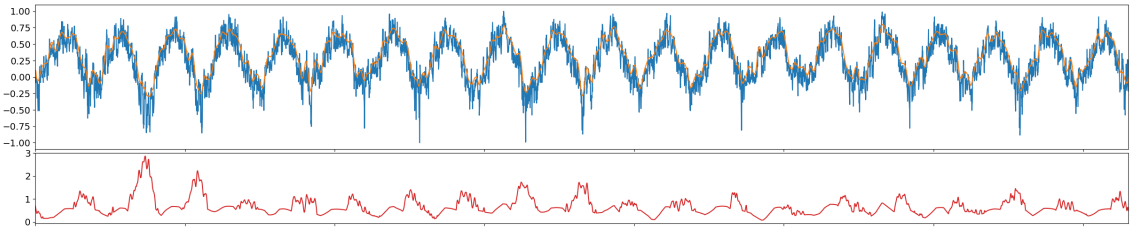


(a) Test 2 error plots and reconstructed signal. The blue line shows the real signal, the orange line shows the reconstructed signal, and the red line shows the error plot.

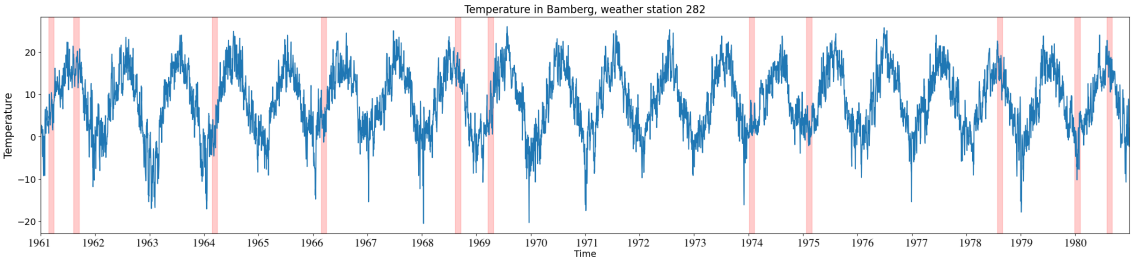


(b) Test 2 anomalies plot. The blue line represents the real signal, while the red regions indicate anomaly regions.

Figure 38: The results of applying the TadGAN method to daily average pressure measurements from 01.01.2001 to 31.12.2020. The reconstructed error was a multiplication of point-wise difference and C_x critic error.

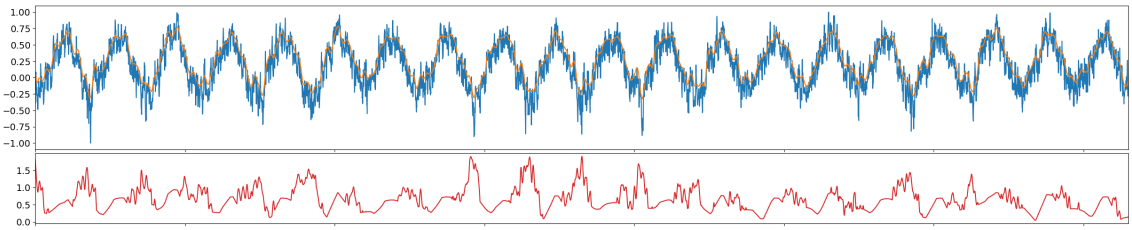


(a) Test 1 error plots and reconstructed signal. The blue line shows the real signal, the orange line shows the reconstructed signal, and the red line shows the error plot.

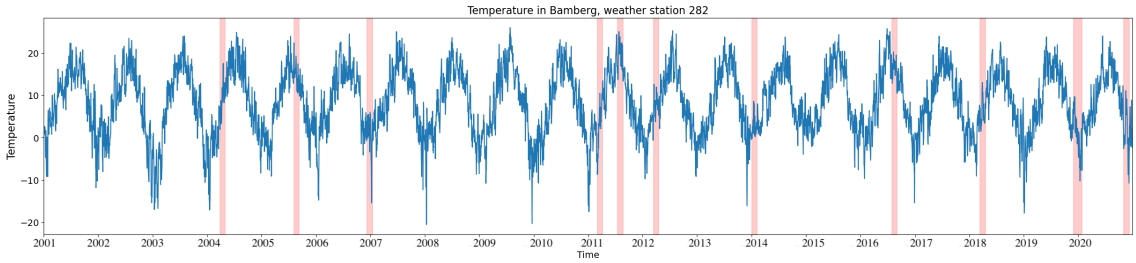


(b) Test 1 anomalies plot. The blue line represents the real signal, while the red regions indicate anomaly regions.

Figure 39: The results of applying the TadGAN method to daily average temperature measurements from 01.01.1961 to 31.12.1980. The reconstructed error was a convex combination of area difference and C_x critic error.

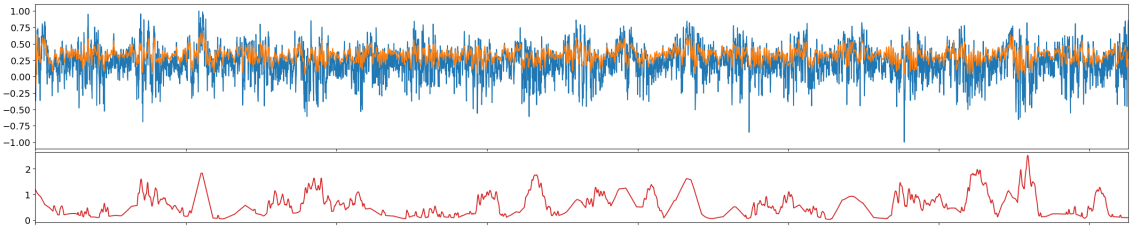


(a) Test 2 error plots and reconstructed signal. The blue line shows the real signal, the orange line shows the reconstructed signal, and the red line shows the error plot.

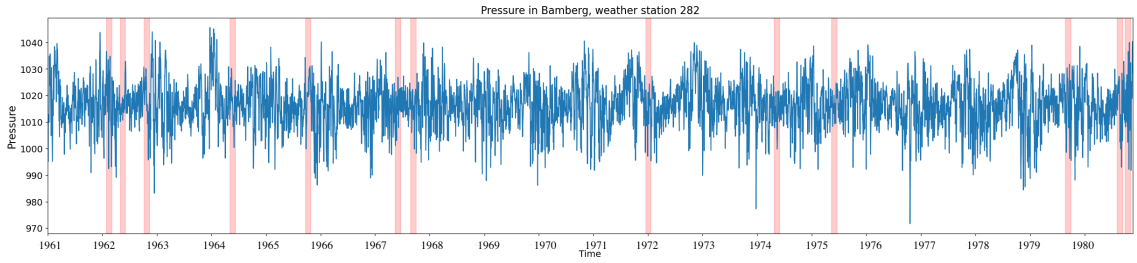


(b) Test 2 anomalies plot. The blue line represents the real signal, while the red regions indicate anomaly regions.

Figure 40: The results of applying the TadGAN method to daily average temperature measurements from 01.01.2001 to 31.12.2020. The reconstructed error was a convex combination of area difference and C_x critic error.

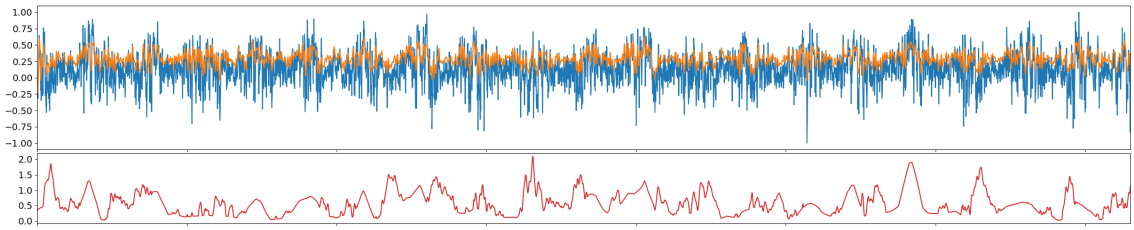


(a) Test 1 error plots and reconstructed signal. The blue line shows the real signal, the orange line shows the reconstructed signal, and the red line shows the error plot.

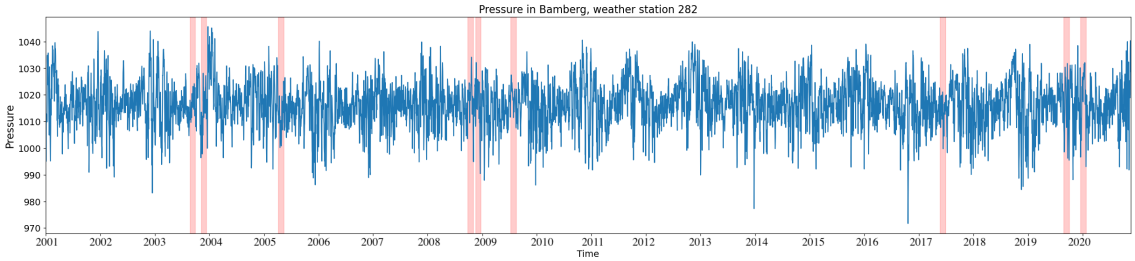


(b) Test 1 anomalies plot. The blue line represents the real signal, while the red regions indicate anomaly regions.

Figure 41: The results of applying the TadGAN method to daily average pressure measurements from 01.01.1961 to 31.12.1980. The reconstructed error was a convex combination of area difference and C_x critic error.

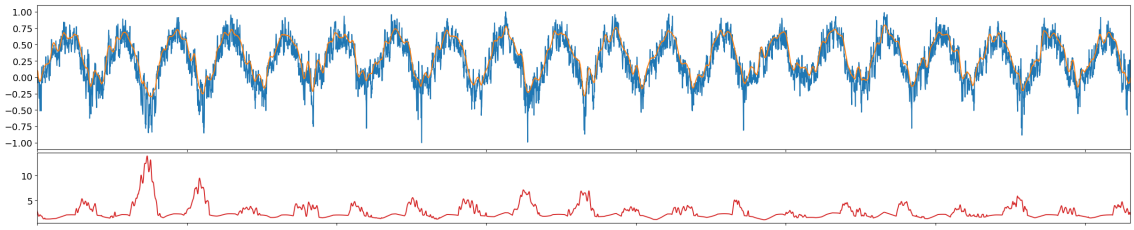


(a) Test 2 error plots and reconstructed signal. The blue line shows the real signal, the orange line shows the reconstructed signal, and the red line shows the error plot.

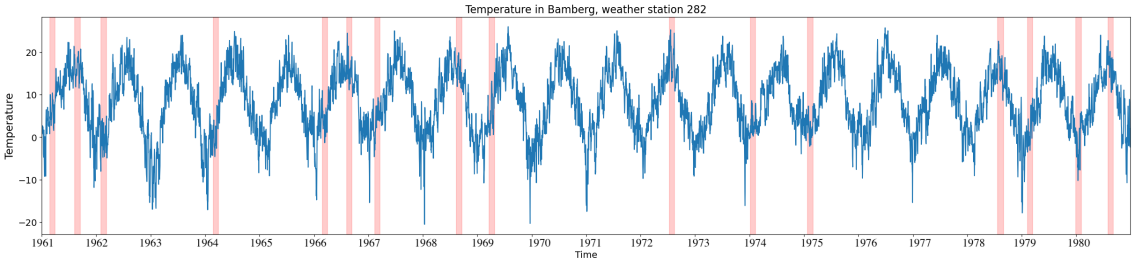


(b) Test 2 anomalies plot. The blue line represents the real signal, while the red regions indicate anomaly regions.

Figure 42: The results of applying the TadGAN method to daily average pressure measurements from 01.01.2001 to 31.12.2020. The reconstructed error was a convex combination of area difference and C_x critic error.

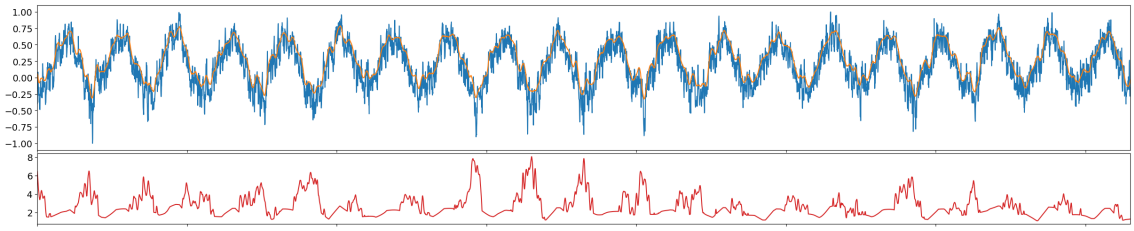


(a) Test 1 error plots and reconstructed signal. The blue line shows the real signal, the orange line shows the reconstructed signal, and the red line shows the error plot.

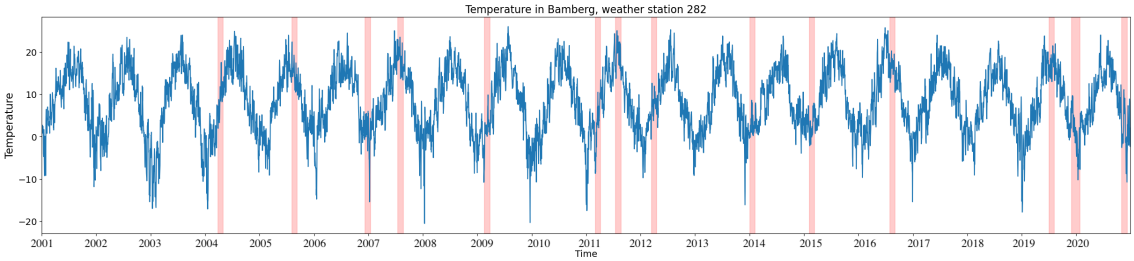


(b) Test 1 anomalies plot. The blue line represents the real signal, while the red regions indicate anomaly regions.

Figure 43: The results of applying the TadGAN method to daily average temperature measurements from 01.01.1961 to 31.12.1980. The reconstructed error was a multiplication of area difference and C_x critic error.

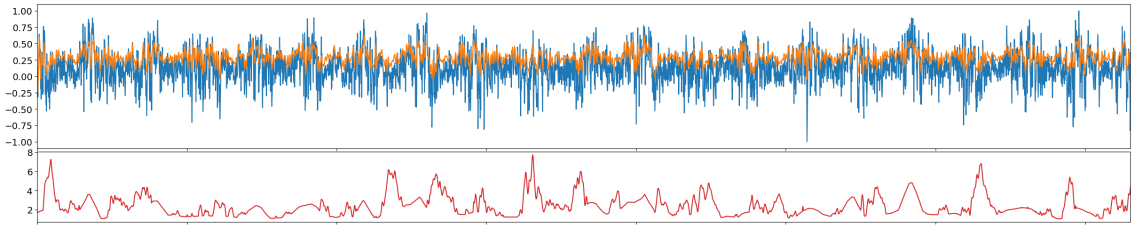


(a) Test 2 error plots and reconstructed signal. The blue line shows the real signal, the orange line shows the reconstructed signal, and the red line shows the error plot.

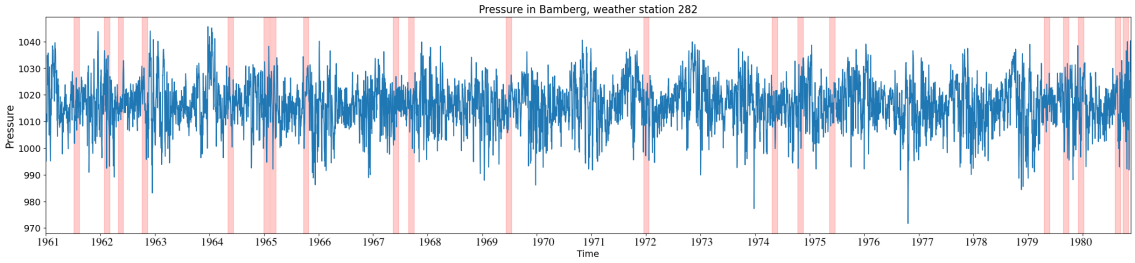


(b) Test 2 anomalies plot. The blue line represents the real signal, while the red regions indicate anomaly regions.

Figure 44: The results of applying the TadGAN method to daily average temperature measurements from 01.01.2001 to 31.12.2020. The reconstructed error was a multiplication of area difference and C_x critic error.

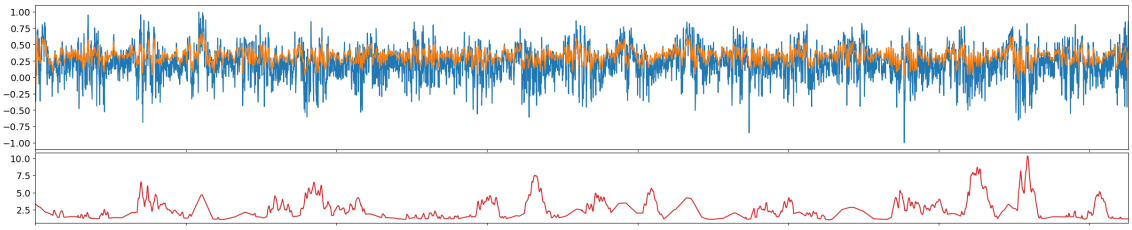


(a) Test 1 error plots and reconstructed signal. The blue line shows the real signal, the orange line shows the reconstructed signal, and the red line shows the error plot.

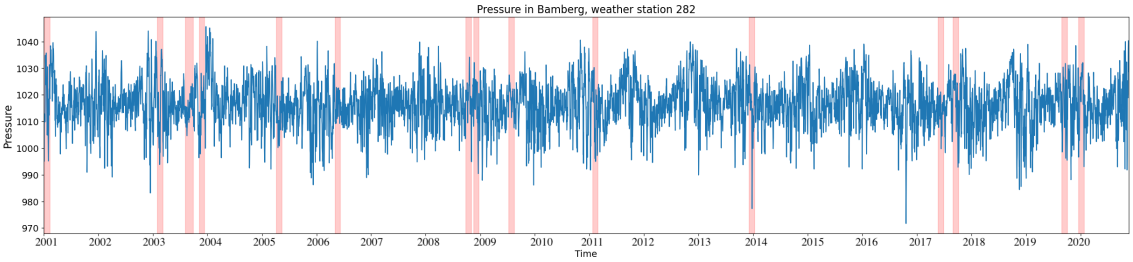


(b) Test 1 anomalies plot. The blue line represents the real signal, while the red regions indicate anomaly regions.

Figure 45: The results of applying the TadGAN method to daily average pressure measurements from 01.01.1961 to 31.12.1980. The reconstructed error was a multiplication of area difference and C_x critic error.

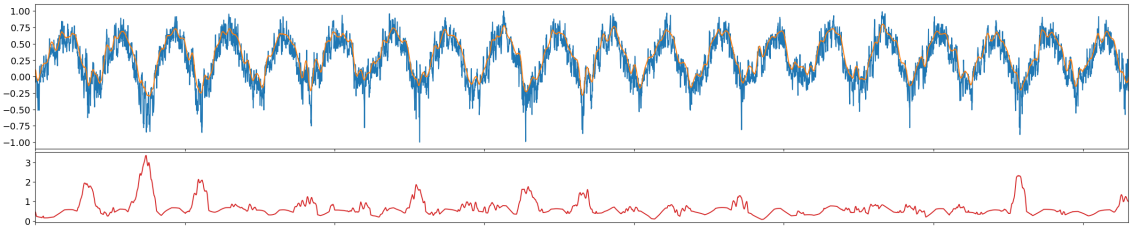


(a) Test 2 error plots and reconstructed signal. The blue line shows the real signal, the orange line shows the reconstructed signal, and the red line shows the error plot.

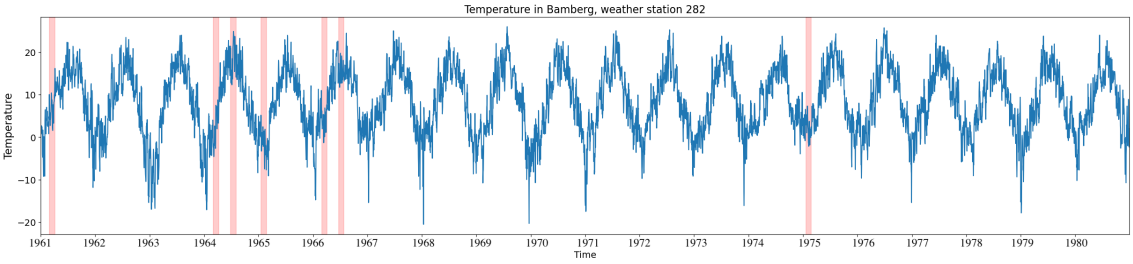


(b) Test 2 anomalies plot. The blue line represents the real signal, while the red regions indicate anomaly regions.

Figure 46: The results of applying the TadGAN method to daily average pressure measurements from 01.01.2001 to 31.12.2020. The reconstructed error was a multiplication of area difference and C_x critic error.

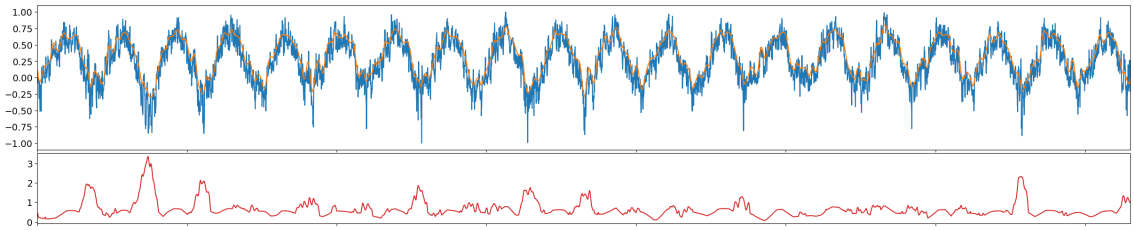


(a) Test 1 error plots and reconstructed signal. The blue line shows the real signal, the orange line shows the reconstructed signal, and the red line shows the error plot.

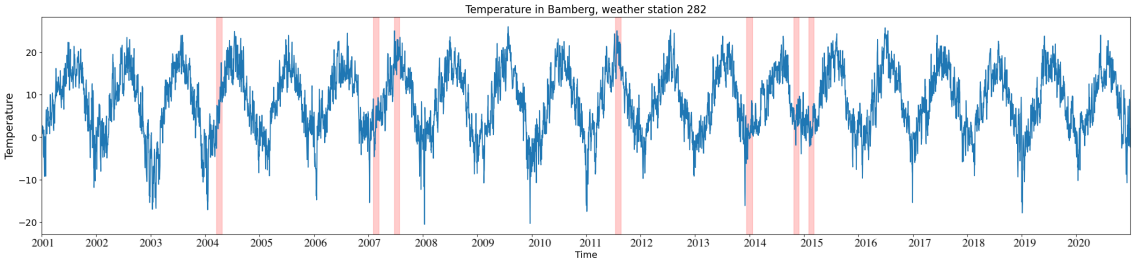


(b) Test 1 anomalies plot. The blue line represents the real signal, while the red regions indicate anomaly regions.

Figure 47: The results of applying the TadGAN method to daily average temperature measurements from 01.01.1961 to 31.12.1980. The reconstructed error was a convex combination of DTW and C_x critic error.

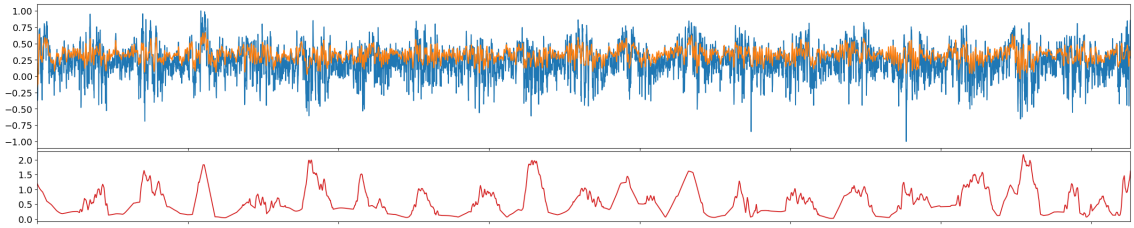


(a) Test 2 error plots and reconstructed signal. The blue line shows the real signal, the orange line shows the reconstructed signal, and the red line shows the error plot.

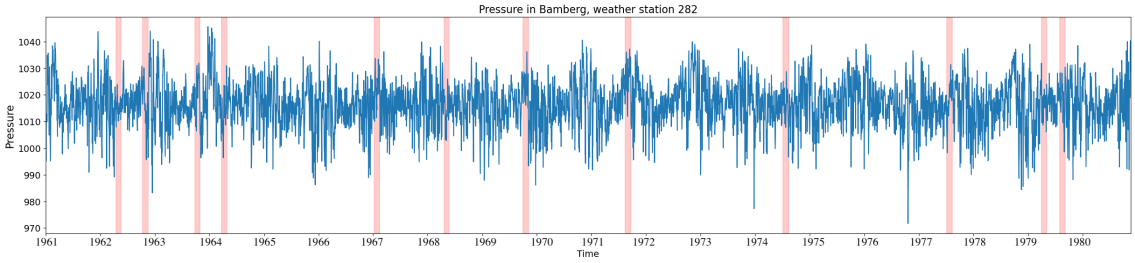


(b) Test 2 anomalies plot. The blue line represents the real signal, while the red regions indicate anomaly regions.

Figure 48: The results of applying the TadGAN method to daily average temperature measurements from 01.01.2001 to 31.12.2020. The reconstructed error was a convex combination of DTW and C_x critic error.

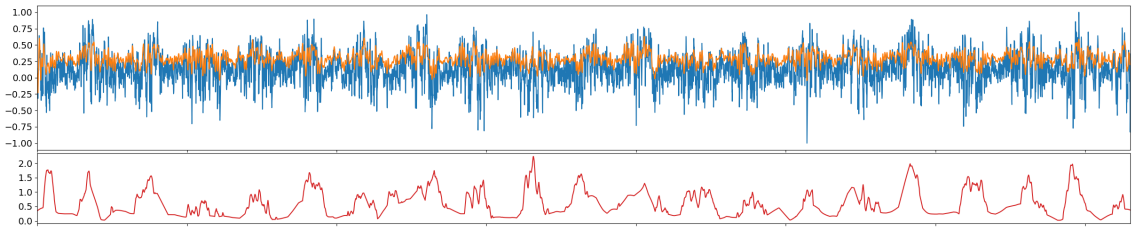


(a) Test 1 error plots and reconstructed signal. The blue line shows the real signal, the orange line shows the reconstructed signal, and the red line shows the error plot.

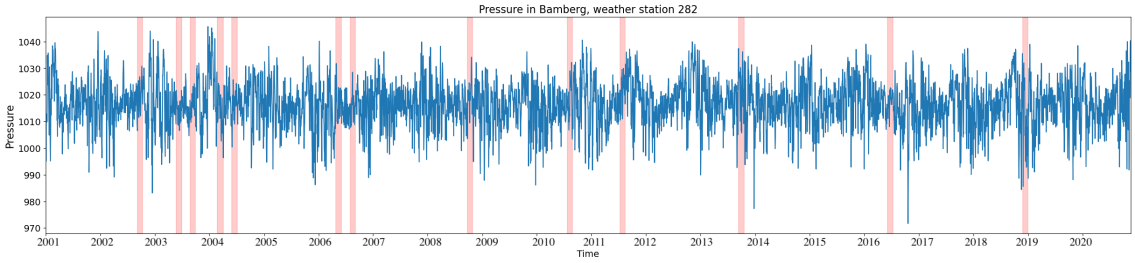


(b) Test 1 anomalies plot. The blue line represents the real signal, while the red regions indicate anomaly regions.

Figure 49: The results of applying the TadGAN method to daily average pressure measurements from 01.01.1961 to 31.12.1980. The reconstructed error was a convex combination of DTW and C_x critic error.

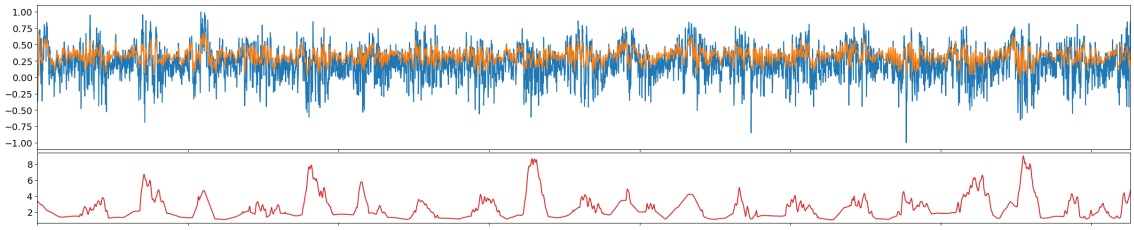


(a) Test 2 error plots and reconstructed signal. The blue line shows the real signal, the orange line shows the reconstructed signal, and the red line shows the error plot.

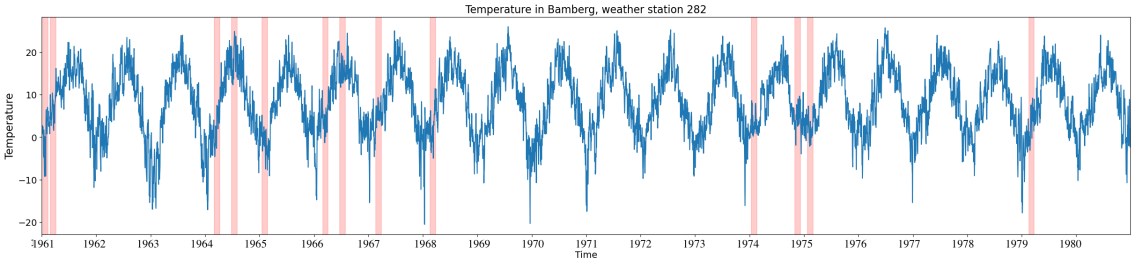


(b) Test 2 anomalies plot. The blue line represents the real signal, while the red regions indicate anomaly regions.

Figure 50: The results of applying the TadGAN method to daily average pressure measurements from 01.01.2001 to 31.12.2020. The reconstructed error was a convex combination of DTW and C_x critic error.

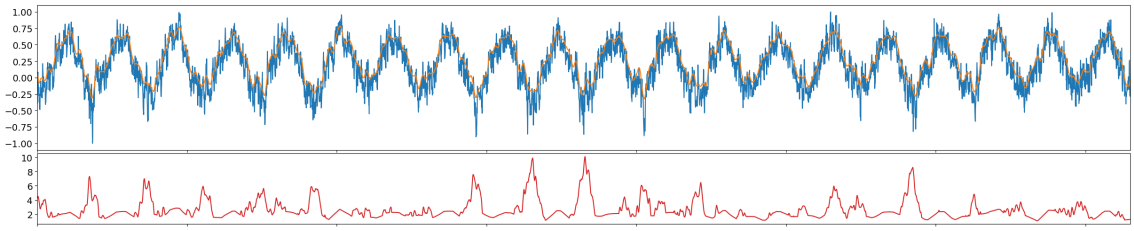


(a) Test 1 error plots and reconstructed signal. The blue line shows the real signal, the orange line shows the reconstructed signal, and the red line shows the error plot.

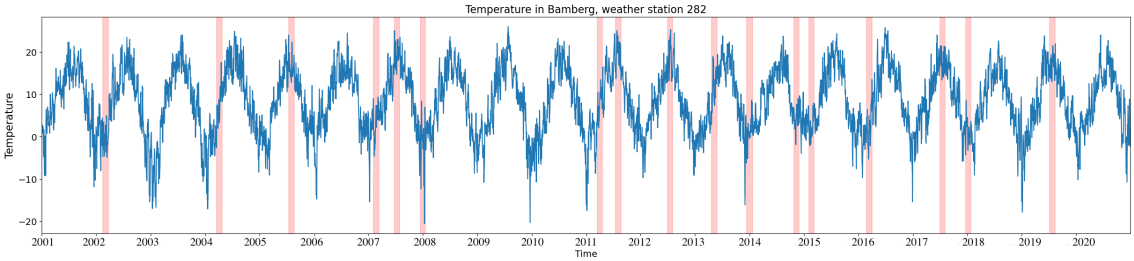


(b) Test 1 anomalies plot. The blue line represents the real signal, while the red regions indicate anomaly regions.

Figure 51: The results of applying the TadGAN method to daily average temperature measurements from 01.01.1961 to 31.12.1980. The reconstructed error was a multiplication of DTW and C_x critic error.

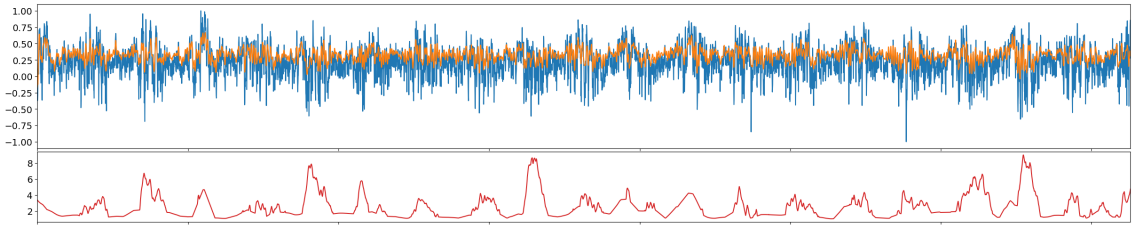


(a) Test 2 error plots and reconstructed signal. The blue line shows the real signal, the orange line shows the reconstructed signal, and the red line shows the error plot.

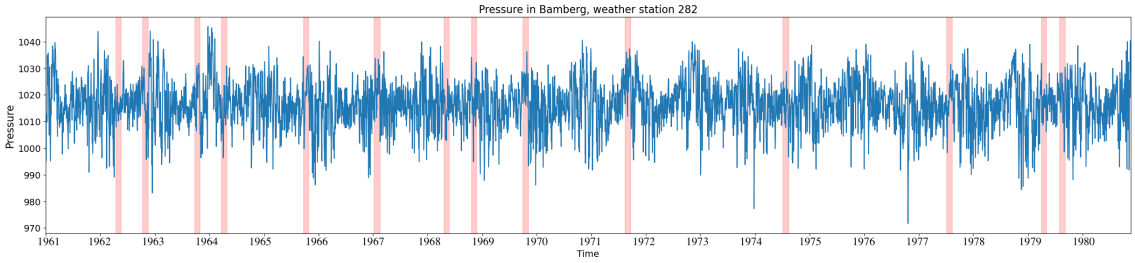


(b) Test 2 anomalies plot. The blue line represents the real signal, while the red regions indicate anomaly regions.

Figure 52: The results of applying the TadGAN method to daily average temperature measurements from 01.01.2001 to 31.12.2020. The reconstructed error was a multiplication of DTW and C_x critic error.

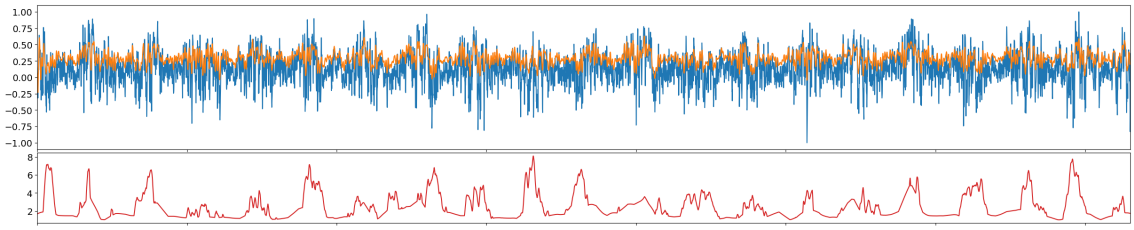


(a) Test 1 error plots and reconstructed signal. The blue line shows the real signal, the orange line shows the reconstructed signal, and the red line shows the error plot.

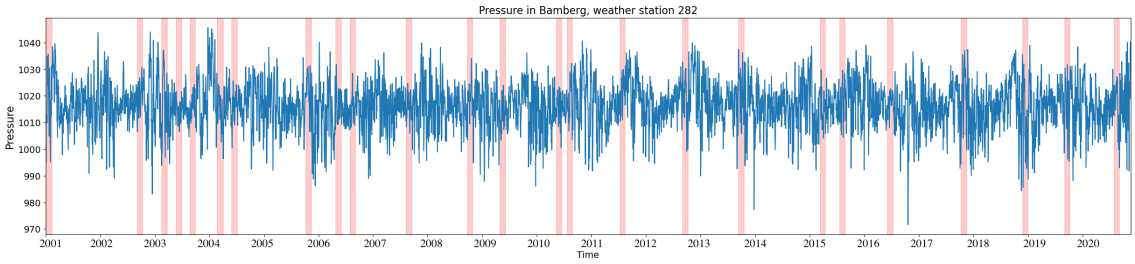


(b) Test 1 anomalies plot. The blue line represents the real signal, while the red regions indicate anomaly regions.

Figure 53: The results of applying the TadGAN method to daily average pressure measurements from 01.01.1961 to 31.12.1980. The reconstructed error was a multiplication of DTW and C_x critic error.



(a) Test 2 error plots and reconstructed signal. The blue line shows the real signal, the orange line shows the reconstructed signal, and the red line shows the error plot.



(b) Test 2 anomalies plot. The blue line represents the real signal, while the red regions indicate anomaly regions.

Figure 54: The results of applying the TadGAN method to daily average pressure measurements from 01.01.2001 to 31.12.2020. The reconstructed error was a multiplication of DTW and C_x critic error.

Bibliography

Faroudja Abid and Latifa Hamami. A survey of neural network based automated systems for human chromosome classification. *Artificial Intelligence Review*, 49: 41–56, 2018.

Oludare Isaac Abiodun, Aman Jantan, Abiodun Esther Omolara, Kemi Victoria Dada, Nachaat AbdElatif Mohamed, and Humaira Arshad. State-of-the-art in artificial neural network applications: A survey. *Heliyon*, 4(11), 2018.

Shalamu Abudu, Chun-liang Cui, James Phillip King, and Kaiser Abudukadeer. Comparison of performance of statistical models in forecasting monthly streamflow of kizil river, china. *Water Science and Engineering*, 3(3):269–281, 2010.

Subutai Ahmad, Alexander Lavin, Scott Purdy, and Zuha Agha. Unsupervised real-time anomaly detection for streaming data. *Neurocomputing*, 262:134–147, 2017.

Hirotugu Akaike. A new look at the statistical model identification. *IEEE transactions on automatic control*, 19(6):716–723, 1974.

Haneen Alabdulrazzaq, Mohammed N Alenezi, Yasmeen Rawajfih, Bareeq A Alghannam, Abeer A Al-Hassan, and Fawaz S Al-Anzi. On the accuracy of arima based prediction of covid-19 spread. *Results in Physics*, 27:104509, 2021.

Sarah Alnegheimish, Dongyu Liu, Carles Sala, Laure Berti-Equille, and Kalyan Veeramachaneni. Sintel: A machine learning framework to extract insights from signals. In *Proceedings of the 2022 International Conference on Management of Data*, SIGMOD '22, page 1855–1865. Association for Computing Machinery, 2022. doi: 10.1145/3514221.3517910.

Ibrahim Alrashdi, Ali Alqazzaz, Esam Aloufi, Raed Alharthi, Mohamed Zohdy, and Hua Ming. Ad-iot: Anomaly detection of iot cyberattacks in smart city using machine learning. In *2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 0305–0310. IEEE, 2019.

Md Abul Bashar and Richi Nayak. Tanogan: Time series anomaly detection with generative adversarial networks. In *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1778–1785. IEEE, 2020.

Ane Blázquez-García, Angel Conde, Usue Mori, and Jose A Lozano. A review on outlier/anomaly detection in time series data. *ACM Computing Surveys (CSUR)*, 54(3):1–33, 2021.

George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

Mohammad Braei and Sebastian Wagner. Anomaly detection in univariate time-series: A survey on the state-of-the-art. *arXiv preprint arXiv:2004.00433*, 2020.

Leo Breiman. Statistical modeling: The two cultures (with comments and a rejoinder by the author). *Statistical science*, 16(3):199–231, 2001.

Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. Lof: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 93–104, 2000.

Ilja N Bronstein, Juraj Hromkovic, Bernd Luderer, Hans-Rudolf Schwarz, Jochen Blath, Alexander Schied, Stephan Dempe, Gert Wanka, and Siegfried Gottwald. *Taschenbuch der mathematik*, volume 1. Springer-Verlag, 2012.

Eoin Brophy, Zhengwei Wang, Qi She, and Tomás Ward. Generative adversarial networks in time series: A systematic literature review. *ACM Computing Surveys*, 55(10):1–31, 2023.

Kung-Sik Chan and Jonathan D Cryer. *Time series analysis with applications in R*. Springer, 2008.

Ngai Hang Chan. *Time series: applications to finance*. John Wiley & Sons, 2004.

Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):1–58, 2009.

Stéphane Crépey, Noureddine Lehdili, Nisrine Madhar, and Maud Thomas. Anomaly detection in financial time series by principal component analysis and neural networks. *Algorithms*, 15(10):385, 2022.

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.

David A Dickey and Wayne A Fuller. Distribution of the estimators for autoregressive time series with a unit root. *Journal of the American statistical association*, 74(366a):427–431, 1979.

Rémi Domingues, Maurizio Filippone, Pietro Michiardi, and Jihane Zouaoui. A comparative evaluation of outlier detection algorithms: Experiments and analyses. *Pattern recognition*, 74:406–421, 2018.

Stephen HC DuToit, A Gert W Steyn, and Rolf H Stumpf. *Graphical exploratory data analysis*. Springer Science & Business Media, 2012.

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.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *arXiv preprint arXiv:1406.2661*, 2014.

Cyril Goutte and Eric Gaussier. A probabilistic interpretation of precision, recall and f-score, with implication for evaluation. In *European conference on information retrieval*, pages 345–359. Springer, 2005.

Jie Gui, Zhenan Sun, Yonggang Wen, Dacheng Tao, and Jieping Ye. A review on generative adversarial networks: Algorithms, theory, and applications. *IEEE transactions on knowledge and data engineering*, 35(4):3313–3332, 2021.

James D. Hamilton. *Time Series Analysis*. Princenton university press, 1994.

Charles Harvie and Mosayeb Pahlavani. Testing for structural breaks in the korean economy 1980-2005: An application of the innovational outlier and additive outlier models. 02 2006.

Douglas M Hawkins. Multivariate outlier detection. In *Identification of outliers*, pages 104–114. Springer, 1980.

Sepp Hochreiter and Jürgen Schmidhuber. Lstm can solve hard long time lag problems. *Advances in neural information processing systems*, 9, 1996.

Serkan Kiranyaz, Onur Avci, Osama Abdeljaber, Turker Ince, Moncef Gabbouj, and Daniel J Inman. 1d convolutional neural networks and applications: A survey. *Mechanical systems and signal processing*, 151:107398, 2021.

Denis Kwiatkowski, Peter CB Phillips, Peter Schmidt, and Yongcheol Shin. Testing the null hypothesis of stationarity against the alternative of a unit root: How sure are we that economic time series have a unit root? *Journal of econometrics*, 54 (1-3):159–178, 1992.

Rikard Laxhammar. *Conformal Anomaly Detection. Detecting abnormal trajectories in surveillance applications*. PhD thesis, University of Skoevde, 1995. URL `https://www.diva-portal.org/smash/get/diva2:690997/FULLTEXT02.pdf`.

Teema Leangarun, Poj Tangamchit, and Suttipong Thajchayapong. Stock price manipulation detection using generative adversarial networks. In *2018 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 2104–2111. IEEE, 2018.

Dan Li, Dacheng Chen, Jonathan Goh, and See-kiong Ng. Anomaly detection with generative adversarial networks for multivariate time series. *arXiv preprint arXiv:1809.04758*, 2018.

Dan Li, Dacheng Chen, Baihong Jin, Lei Shi, Jonathan Goh, and See-Kiong Ng. Mad-gan: Multivariate anomaly detection for time series data with generative adversarial networks. In *International conference on artificial neural networks*, pages 703–716. Springer, 2019.

Zijian Niu, Ke Yu, and Xiaofei Wu. Lstm-based vae-gan for time-series anomaly detection. *Sensors*, 20(13):3738, 2020.

Sebastian Schmidl, Phillip Wenig, and Thorsten Papenbrock. Anomaly detection in time series: a comprehensive evaluation. *Proceedings of the VLDB Endowment*, 15(9):1779–1797, 2022.

Alex Shenfield and Martin Howarth. A novel deep learning model for the detection and identification of rolling element-bearing faults. *Sensors*, 20(18):5112, 2020.

Robert H. Shumway and David S. Stoffer. *Time Series Analysis and Its Applications*. Springer, 2000.

Robert H Shumway, David S Stoffer, and David S Stoffer. *Time series analysis and its applications*, volume 3. Springer, 2000.

Yuqiang Sun, Lei Peng, Huiyun Li, and Min Sun. Exploration on spatiotemporal data repairing of parking lots based on recurrent gans. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 467–472. IEEE, 2018.

Jo-Anne Ting, Evangelos Theodorou, and Stefan Schaal. A kalman filter for robust outlier detection. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1514–1519. IEEE, 2007.

Ruey S Tsay. Outliers, level shifts, and variance changes in time series. *Journal of forecasting*, 7(1):1–20, 1988.

Wei-Tung Wang, Yi-Leh Wu, Cheng-Yuan Tang, and Maw-Kae Hor. Adaptive density-based spatial clustering of applications with noise (dbscan) according to data. In *2015 International Conference on Machine Learning and Cybernetics (ICMLC)*, volume 1, pages 445–451. IEEE, 2015.

Jinsung Yoon, Daniel Jarrett, and Mihaela Van der Schaar. Time-series generative adversarial networks. *Advances in neural information processing systems*, 32, 2019.

Wen Yu, Haibo He, and Nian Zhang. *Advances in Neural Networks-ISNN 2009: 6th International Symposium on Neural Networks, ISNN 2009 Wuhan, China, May 26-29, 2009 Proceedings, Part II*, volume 5552. Springer, 2009.

Declaration of Authorship

Ich erkläre hiermit gemäß §9 Abs. 12 APO, dass ich die vorstehende Abschlussarbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Des Weiteren erkläre ich, dass die digitale Fassung der gedruckten Ausfertigung der Abschlussarbeit ausnahmslos in Inhalt und Wortlaut entspricht und zur Kenntnis genommen wurde, dass diese digitale Fassung einer durch Software unterstützten, anonymisierten Prüfung auf Plagiate unterzogen werden kann.

Place, Date

Signature