Supporting Experts in Detecting and Interpreting Anomalies in Time Series

Exploring Data Science Approaches for the Monitoring of Hydraulic Test Benches

Deniz Neufeld

Submitted in partial fulfillment of the requirements for the degree of
Doctor of Natural Sciences (Dr. rer. nat.)
—
Faculty for Information Systems and Applied Computer Sciences
University of Bamberg

Supervisor and Reviewer:
Prof. Dr. Ute Schmid

Second Reviewer:
Prof. Dr. Daniela Nicklas

Head of Examining Board:
Prof. Dr. Diedrich Wolter

Bamberg 2024

Submitted in partial fulfillment of the requirements for the degree of Doctor of Natural Sciences (Dr. rer. nat.), Faculty for Information Systems and Applied Computer Sciences.

Supervisor and Reviewer:
Prof. Dr. Ute Schmid

Second Reviewer:
Prof. Dr. Daniela Nicklas

Head of Examining Board:
Prof. Dr. Diedrich Wolter

Defended on Dec. 8th, 2023.

URN: urn:nbn:de:bvb:473-irb-954138
DOI: https://doi.org/10.20378/irb-95413

Abstract

Hydraulic systems are important in the functioning of everyday life, as well as critical infrastructure by driving various systems such as automotive, aircrafts, and construction machines. To assure their functionality according to specification over their complete lifetime, reliability tests are conducted in test benches to check a representative number of samples for long amounts of time. During the tests, physical system inputs and outputs such as pressures, electric currents and voltages are recorded for analysis. The goal of this thesis is to research anomaly detection methods based on the recorded data to support root cause analysis and reduce testing time. Since the first decision makers in case of an error are the responsible engineers and technicians, it is important to focus on methods that are both robust and easily understandable for non-experts of data science.

Several challenges stem from the given problem. With the start of a test bench, there is often no prior measurement data available for model training, and samples are not repeated after a successful test run. Furthermore, there are various kinds of signals with di erent properties, from digital bus system signals to fluid flow or electrical currents. If a complete system is assessed, di erent failure modes can occur based on di erent sub-components. This leads to many possible failure constellations and therefore a large number of relevant features.

There are multiple fields of research which are related to this problem, ranging from the general field of time series analysis to more specific condition monitoring research in the domain of engineering. This means there exist similar problem types in state of the art, but also limited closely related work. The following work not only focuses on research on the anomaly detection on hydraulic test benches per se, but also investigates data visualization, anomaly detection in physical systems, multivariate time series anomaly detection, and vibration analysis.

The original contribution to knowledge of this thesis are advances in multiple aspects of anomaly detection with a focus on hydraulic test benches. First a collection of methods for the visualization of multivariate, repeating time series is provided. It supports engineers in viewing data and detecting anomalies visually by aligning the signals along the time and the amplitude axis. From this, this thesis examines a model-based method for a digital twin of the tested system using recurrent neural networks. Challenges with this approach are shown and their root cause described in depth. Additionally, data-based methods are examined: One, an unsupervised, statistical method is developed for anomaly detection on multivariate, periodical data in the time domain, as well as its robustness and limits of this method towards concept drift are investigated. It is further shown how the results of the method can be visualized for root-cause analysis. Second, a supervised approach is followed on vibrational

data using convolutional neural networks (CNNs). For this, preprocessing in the frequency-domain its influence on model performance is researched. Due to the black box nature of CNNs, an explainable artificial intelligence method is developed to make the relevant features of the data in the frequency domain interpretable for engineers and system experts. The XAI method is verified in quantitatively using an accessible data set specifically designed for this task.

Based on the shown research, this thesis presents possibilities for future work, specifically with a focus on the enormous amounts of data collected. During a test bench run, millions of time series can be collected, but the visualization and anomaly detection methods developed are not yet perfected towards this fact. Due to the size of the data, dealing with the data and the results of the anomaly detection methods must become more efficient as well, be it for displaying, labeling or for judging the output of the algorithms. While the methods shown can be used on subsets of the data, an important aspect of future work is the clustering of data to reduce the amount of data to support inspection and labeling. Additionally, the higher-level visualizations with a drill-down functionality need to be developed as well to guide users towards relevant information.

Zusammenfassung

Hydraulische Systeme sind erfüllen wichtige Funktionen im Alltag und in kritischen Infrastrukturen, da sie Teil verschiedene Systeme wie Kraftfahrzeuge, Flugzeuge und Baumaschinen sind. Um ihre Funktion über ihre gesamte Lebensdauer hinweg zu garantieren, werden Zuverlässigkeitstests in Prüfständen durchgeführt, bei denen eine repräsentative Anzahl von Komponenten über einen langen Zeitraum getestet werden. Während dieser Tests werden physikalische System Ein- und -ausgänge wie hydraulische Drücke, elektrische Ströme und Spannungen zur späteren Analyse aufgezeichnet. Ziel der vorliegenden Arbeit ist es, Methoden zur Erkennung von Anomalien auf der Grundlage der aufgezeichneten Daten zu erforschen, um die Ursachenanalyse im Fehlerfall zu erleichtern und Prüfzeiten zu verkürzen. Da die ersten Entscheidungsträger im Falle eines Fehlers die verantwortlichen Ingenieure und Techniker sind, stehen Methoden im Fokus, die sowohl robust als auch für Nicht-Experten der Datenwissenschaft leicht verständlich sind.

Aus dieser Problemstellung ergeben sich mehrere Herausforderungen. Beim Start eines Prüfstands stehen oft keine vorherigen Messdaten für das Trainieren von Modellen zur Verfügung, und die Prüfläufe werden nach einem erfolgreichen Durchlauf nicht wiederholt. Außerdem gibt es unterschiedliche Arten von Signalen, z. B. digitale Bussignale, Durchflussraten oder elektrische Ströme. Wenn ein komplettes System getestet wird, können verschiedene Fehlermodi abhängig von verschiedenen Unterkomponenten auftreten. Dies führt zu vielen möglichen Fehlerkonstellationen und damit zu einer großen Anzahl von möglicherweise relevanten Merkmalen.

Es gibt mehrere Forschungsbereiche, die mit diesem Problem in Zusammenhang stehen: vom allgemeinen Bereich der Zeitreihenanalyse bis hin zur spezifischeren Condition Monitoring, wie es im Ingenieurswesen verwendet wird. Das bedeutet, dass es im Stand der Technik mehrere ähnliche Problemtypen gibt, aber auch sehr wenige eng verwandte Arbeiten. Diese Arbeit konzentriert sich nicht nur auf die Forschung zur Erkennung von Anomalien an Hydraulikprüfständen an sich, sondern untersucht auch die Datenvisualisierung, die Erkennung von Anomalien in physikalischen Systemen, die Detektion von multivariaten Zeitreihenanomalien und die Schwingungsanalyse.

Der Wissensbeitrag dieser Arbeit sind Fortschritte in mehreren Aspekten der Anomalieerkennung mit Schwerpunkt auf Hydraulikprüfstände. Zunächst wird eine Methodensammlung für die Visualisierung multivariater, sich wiederholender Zeitreihen vorgestellt. Sie beinhaltet die Ausrichtung von Signales entlang der Zeit- und Amplitudenachse, um Anwender (Techniker und Ingenieure) bei der Betrachtung von Daten und der visuellen Erkennung von Anomalien zu unterstützen. Darauf aufbauend

wird in dieser Arbeit eine modellbasierte Methode für einen digitalen Zwilling des getesteten Systems mithilfe rekurrenter neuronaler Netze untersucht. Die Herausforderungen dieses Ansatzes werden aufgezeigt und ihre Ursachen eingehend beschrieben. Deshalb werden anschließend datenbasierte Methoden erforscht: Zum einen wird eine unüberwachte, statistische Methode zur Erkennung von Anomalien auf multivariaten, periodischen Daten im Zeitbereich entwickelt, sowie deren Robustheit und die Grenzen dieser Methode bei Konzeptdrift untersucht. Darüber hinaus wird gezeigt, wie die Ergebnisse der Methode für die Ursachenanalyse visualisiert werden können. Anschließend wird ein überwachter Ansatz für Schwingungsdaten unter Verwendung von Convolutional Neural Networks (CNNs) verfolgt. Hierfür wird der Einfluss der Vorverarbeitung im Frequenzbereich auf die Modellleistung untersucht. Aufgrund der Black-Box-Natur von CNNs wird eine erklärbare Methode der künstlichen Intelligenz (explainable AI, XAI) entwickelt, um die relevanten Merkmale der Daten im Frequenzbereich für Ingenieure und Systemexperten interpretierbar zu machen. Die XAI-Methode wird anhand eines speziell für diese Aufgabe entwickelten Datensatzes quantitativ verifiziert.

Basierend auf den gezeigten Forschungsergebnissen werden mehrere Möglichkeiten für zukünftige Arbeiten vorgestellt, insbesondere im Hinblick auf die enormen Datenmengen, die bei Testläufen anfallen können. Während eines Prüflaufs können Millionen von Zeitreihen gesammelt werden, aber die erarbeiteten Visualisierungs- und Anomalieerkennungsmethoden sind noch nicht auf diese Tatsache optimiert. Durch die Größe der erfassten Datenmengen muss auch der Umgang mit den Daten und den Ergebnissen möglicher Methoden zur Anomalieerkennung effizienter werden, sei es für das Labeling oder für die Beurteilung der Ergebnisse der Algorithmen. Während die gezeigten Methoden auf Teilmengen der Daten angewendet werden können, ist ein wichtiger Aspekt zukünftiger Arbeit das Clustering von Daten. Dadurch kann die Menge an Daten reduziert werden, die inspiziert und manuell beschriftet werden muss. Darüber hinaus muss die Visualisierung solcher Datenmengen effizienter werden, was bedeutet, dass für groß angelegte Tests auch übergeordnete Visualisierungen mit Drill-Down-Funktionalität entwickelt werden müssen.

Acknowledgments

I am very grateful to Professor Ute Schmid from the University of Bamberg for her guidance, feedback, and patience during my PhD thesis. Furthermore, I thank her as well as Prof. Nicklas and Prof. Diedrich Volter for their feedback during my PhD colloquium and on the thesis itself and their suggestions for improvement. They had a major influence on the structure and content of this thesis.

My sincere thanks also go to Dr. Achim Romer, who supervised and supported this thesis by providing me with a working environment that allowed efficient research and by giving me a lot of freedom in the direction of my research, while at the same time providing me with valuable advice on possible research topics, the topic itself, and possible boundary conditions.

My research would not be possible without the funding and support of Robert Bosch GmbH in Abstatt. I also want to thank my numerous colleagues at the company for supporting my research by introducing me to the technical background on hardware, their day-to-day work, and supplying me with feedback on my work.

On my university's side, I had the pleasure of working with a supportive group of PhD students in the Cognitive Systems group in Bamberg, which led to very insightful discussions about other, diverse research topics and insights for my own research. Special thanks go to Oliver Mey for the collaboration on our paper and to Professor Ute Schmid for leading to our exchange. The collaboration and research was very motivating and productive, and I believe that the research results of our joint work are better than the sum of the individual parts would have been.

Finally, I would not have been able to embark on this journey without the support of Jakob and his family while working on my PhD, and during my master's studies before it. I am looking forward to being a more active participant in everyday life soon. Special thanks also go to my friends who have supported me throughout this PhD, especially for their feedback and our working and writing sessions at libraries, via phone during Covid, or at each others' homes. Without them, I would have had to submit this thesis several months later for sure.

Contents

List of Tables

List of Figures

Acronyms

AD Anomaly detection.

AI Artificial Intelligence.

CNN Convolutional Neural Networks.

CPS Cyber Physical System.

DM Data Mining.

DNN Deep Neural Networks.

ECU Electronic Control Unit.

FFT Fast Fourier Transform.

GRU Gated Recurrent Unit.

IoT Internet of Things.

IT Information Technology.

KDDM Knowledge Discovery and Data Mining.

KNN k-nearest neighbor.

LIME Local Interpretable Model-Agnostic Explanations.

LOF Local Outlier Factor.

LSTM Long Short Term Memory.

MAE Mean Absolute Error.

ML Machine Learning.

MSE Mean Squared Error.

ODE Ordinary Di erential Equation.

PHM Prognostics and Health Management.

RN Recurrent Networks.

RNN Recurrent Neural Networks.

RPM Revolutions per minute.

STFT Short-time Fourier Transform.

SVM Support Vector Machine.

XAI Explainable AI.

1 Introduction

Rising processing power and advances in networking technologies have led to increasing data collection and processing applications in the industrial context, supported by increasingly interconnected systems. There are various names for this trend, including the fourth wave of industrial revolution (Industry 4.0), the digitalization of industrialized production, or Big Data applications [1]. The general goal is to raise efficiency, flexibility, and save resources through data-based decision support [2] and to gain further efficiency with the use of artificial intelligence (AI) methods [3]. Still, while most companies confirm the value of Big Data applications, large parts (up to 70%) of recorded data are never used, due to insufficient skills of workers or inconvenient tooling provided [4]. Both reasons depend – at least partially – on the background knowledge of users, which is why it can be assumed that certain user groups tend to struggle more than others with data analysis tasks. An example are engineers specialized in hardware development compared to data scientists. Conversely, in the domain of hardware development and manufacturing, substantial amounts of data are recorded, which could yield added business value [1]. Uses are, among others, anomaly detection, digital models for simulation, and failure prediction [5].

The focus of this thesis is the analysis of data recorded during the testing of hydraulic systems, to detect anomalies as soon as possible. Hydraulic systems convert energy, e.g., from electrical current to pressure per pumps driven by motors. The resulting pneumatic force is controlled and directed using valves [6]. Such systems are versatile since they make it possible to transfer force via fluid in narrow and flexible tubes. Hydraulic systems are applicable in di erent domains, such as brake systems in vehicles, actuators in construction machines, or in plastic injection molding. Hydraulic systems consist of electrical and mechanical components, which are often controlled digitally with electronic control units (ECUs).

From a reliability perspective, this means that various kinds of anomalies and defects can occur in a such systems, e.g., gear failures, bearing defects, leakages, or overheating of electrical components. Reasons for defects are e.g., manufacturing errors, which can lead to errors at the start of the lifetime of a system. Aside from this, excessive load, aging of materials, or corrosion can be the cause of anomalies. Defects can be detected from system measurements by analyzing internal and external properties of a hydraulic system, for example based on input and output pressures, electrical currents and voltages, system vibrations and noise [7], [8].

During system development, it is the focus of reliability engineers to validate that all sub-components will perform as specified over a system's lifetime, even after su ering wear, excessive load, corrosion, or aging

[7]. This is done by testing multiple parts of either the complete system or its sub-components, over extended periods of time using accelerated life testing [9], [10] with test benches. Test benches are programmed to apply the load expected after deployment in a shortened amount of time, simulating the lifetime load of a system under use and thereby aging it faster artificially. During these tests, digital control signals as well as hardware inputs and outputs (e.g., electric currents, voltages, hydraulic pressures) are recorded. This results in large amounts of multivariate, highly sampled time series data. When a system fails prematurely during the test, engineers can analyze this data to find the sub-component that is the root cause [7]. Based on the results, adjustments to the system's design, materials or manufacturing process can be necessary.

During the test bench run, it is the goal of automated anomaly detection to detect changes in a system, or its sub-components based on measured signals as soon as possible. This would o er several benefits. Some failure cases can cause defects in other components, making the root cause analysis more difficult [11]–[14]. Additionally, since the test hardware is expensive and limited, reducing the runtime would bring time and financial savings regarding the use of test bench infrastructure [15]. Shorter test bench runs also mean that system re-design can take place sooner, and a new test can be restarted with a newer, optimized version of the system. Without automation, in the worst case there must be regular measurements and function evaluations by a technician, who removes the test sample from the test bench to check the system's condition.

The development of anomaly detection methods in the context of accelerated testing poses multiple challenges. If the tested components are first-of their-kind prototypes, there is no prior data for the training of models. But, even if data of prior test runs exists, there is the issue of data quality. Mistakes during the setup and programming of the test bench can lead to missing sensor data. After the test bench run, lack of proper IT infrastructure and time pressure often mean that the analysis outcomes of a test run, i.e., failure information of sub-components, are not saved in a digitalized way that is reusable for AI application, which means that the recorded measurements lack meaningful labels. Another problem is the amount of data recorded, which often is so large that it makes the data difficult to visualize and analyze [16]. Here, data science methods with a focus on visualization and data reduction could be of beneficial. Lastly, the aging of the system can cause a concept drift in the recorded data, which must be considered when designing anomaly detection algorithms.

Finally, it is a requirement that the algorithms, or at least their output, be understandable to the decision makers and experts in the testing process, i.e., reliability engineers and test bench technicians. As mentioned before, one essential reason of unused data is the struggle of workers based on their background knowledge and the tooling. Non-understandable methods lead to bad user acceptance and reduces the chance of beneficial influence of an anomaly detection tool or model. This applies not just if the model predicts an anomaly correctly. Turning o the test bench prematurely due to a false prediction of a model would be more detrimental

than having the test run for too long, because then the system test would be stopped before the goal of the test run has been achieved, causing essential failure information to not be recorded.

Still, there are several aspects that can be used when developing data science methods in this context, by choosing proper visualizations and models. In the following work, model-based and data-based approaches are distinguished. Model-based meaning that the system under test is replicated with a digital model, which is to serve as a description of the underlying system and its properties [17]. The amount of deviation between model and reality can then be classified as anomaly. This contrasts with data-based methods, where the data of the system is used as input to a model directly.

Since the test bench is pre-programmed, large parts of the measured signals are there in the shape of periodic time series, which enables visual and statistical comparison [15], [18].

Regarding the missing labeled data, since multiple systems are tested at the same time, one can assume that the average behavior of a system is the normal. Therefore, a significant di erence of one system to all others can be interpreted as abnormal, enabling a semi-supervised approach despite missing labels in the data [15].

Lastly, while some components are changed between test bench runs, others tend to be reused more often, e.g., motors or ball bearings. Therefore, supervised approaches can also be applied. This makes it possible to use a trained classifier to extract further information on a system using explainable AI (XAI) methods (as shown in Chapter 7).

In this work, several aspects of the problem domain of test bench anomaly detection are investigated. First, visualization techniques for test bench data are shown. It is a best practice in data science to first get a visual impression of the given data before designing a model. First, visualization techniques are explored to simplify the detection of anomalies for engineers. Afterwards, a model-based approach using recurrent neural networks is examined, to model the normal system's behavior for anomaly detection. Challenges with this approach will be detected and investigated. Based on this, the subsequent approaches focus on data-based methods. An unsupervised anomaly detection method for periodic, multivariate time series is developed, which can function without prior training data and thresholding. Additionally, this algorithm is shown to detect anomalies if there is a concept drift in the data. While this approach detects anomalies in the time domain, it is not applicable for data in the frequency domain. This is an important topic in anomaly detection for hydraulics, though, for example for motor or ball bearing anomaly detection. Therefore, the next chapter focuses on supervised deep learning models for the data-based anomaly detection of vibration time series, in the frequency domain using Convolutional Neural Networks (CNNs). Since CNNs are Black Box Models and not easily interpretable for human users, a saliency mapping method for vibrational data is developed.

1.1 Research Questions

This work looks at several aspects of the topic of anomaly detection in test benches. This goes from visualization, to modeling of anomalies, to more sophisticated models and feature engineering. Based on this, the following research questions are explored:

- How can data science methods be used for the anomaly detection in test benches?
- Which methods are suitable for which use cases?
- How can the expert (user) interact with and interpret the output of these methods?

1.2 Scientific Contributions

Given the research questions, this thesis investigates multiple various research directions and the following scientific contributions are made:

- Di erent aspects of data science solutions regarding the anomaly detection for physical systems are shown, ranging from data visualization, model-based approaches to data-based approaches for time domain data and frequency domain data.
- A visualization framework for periodic, multivariate time series is developed to simplify the displayed plots for the human eye and minimize redundant information. Given repeating time series patterns, the time series are plotted along one period and o sets along the time-axis and amplitude axis are reduced. Additionally, methods for interactivity of the plots are proposed. The method is presented in Chapter3.
- Chapter 4 explores the modeling of the behavior of complex hydraulic systems using recurrent neural networks. Challenges were found during this research and are then examined. Their root cause is explained using di erent datasets and model constellations.
- A new algorithm for unsupervised anomaly detection for periodic, multivariate time series that is robust towards concept drift is described in Chapter 5. Based on the assumption that the average system behaves normally, the di erence between the individual system and the "normal" is used for anomaly classification. Various distance metrics and anomaly classification algorithms are examined. For evaluation, a data set published by Helwig et al. [8] was adapted and expanded to simulate concept drift. While this method performs well for data in the time domain, experiments also show that features in the frequency domain to not result in meaningful accuracy with this approach and the underlying data set.
- Data features in the frequency domain are relevant for e.g., vibration analysis of rotating systems. This is relevant for rotating components

of mechanical systems, such as gear boxes or ball bearings, in case of e.g., imbalances or damage. Therefore, the influence of spectral transformations (frequency- and order-RPM maps) on classification accuracy of a Convolutional Neural Network (CNN) model was examined. For this and the evaluation of XAI methods in Chapter 6, a data set based on sine curves was developed.

- A new variant of XAI method, called Spectral LIME for frequency and order transformed data was developed in Chapter 7 to show relevant spectra of vibration time series data. It differs from the original method which perturbs the input data with e.g., noise or zeros, Spectral LIME by instead using data of the counter class. Additionally, more consistent explanations were created by applying LIME repeatedly with different segment sizes and numbers and averaging the results.

1.3 Outline

The thesis is organized in the following structure:

- Chapter 2 supplies background information for this thesis. Basics on hydraulic systems and the motivation of the use of hydraulic test benches are described, followed by common maintenance strategies. Additionally, theoretical basics on time series, time series processing, and anomaly detection methods are discussed.
- Chapter 3 presents a collection of visualization strategies for periodic, multivariate time series data.
- Chapter 4 explores the approach and challenges of data-based modeling of hydraulic systems using recurrent neural networks.
- Chapter 5 introduces an algorithm for unsupervised anomaly detection of test bench data for data in the time domain.
- Chapter 6 examines the domain of vibration analysis for rotating systems and the influence of two preprocessing methods on model accuracy.
- Chapter 7 investigates XAI algorithms to make the prior CNNs explainable for expert end users.
- Chapter 8 concludes the thesis, summarizes its results, and supplies a discussion and outlook on future work.

2 State of Research in Anomaly Detection of Measured Signals

In the following, basic information on the background of hydraulic systems is discussed, followed by definitions relevant for AI and data science, particular with regards to time series. Afterwards, a summary of related work on the topic of anomaly detection in time series is provided. Note that this chapter aims to provide a general overview on this topic. Since this thesis examines several distinct aspects, each chapter also will investigate the state of the art relevant to the topics discussed.

2.1 Application Domain

This section supplies information on the functioning of hydraulic systems and reliability analysis in hardware, to show the underlying motivation and challenges of this work.

2.1.1 Hydraulic Systems

Hydraulic systems transfer electric energy into hydraulic pressures and thereby forces. Electric motors actuate pumps which in turn generate hydraulic pressure and fluid flow. In their most basic from, the resulting forces can be described using Pascal's principle: Given a system (shown in Figure 2.1) with two entry planes A_1 and A_2, where respectively the forces F_1 and F_2 are applied, the pressure p in the system can be described using the equation

$$p = \frac{F_1}{A_1} = \frac{F_2}{A_2}, \tag{2.1}$$

which means that small forces applied to a small area can be transformed into large output forces at a larger area.

Hydraulics are used in numerous domains, e.g., automotive, construction machinery, or aerospace engineering. Their advantages – as opposed to electrical drives – are high precision, the repeatability of movement, the high output forces, and the fact that positions can be held even if no energy is applied [6].

Examples of hydraulic components are electrical motors, which actuate pumps (for example gear pumps shown in Figure 2.2a), valves for changing the flow direction of fluids, or reservoirs for fluid accumulation [6]. A system's state can be measured using sensors and thereby supervised, as shown in Figure 2.3. Example signals are electrical currents and voltages, hydraulic pressures, mechanical forces, and control signals, e.g., from an electronic control unit (ECU). Based on the same inputs, the output of two

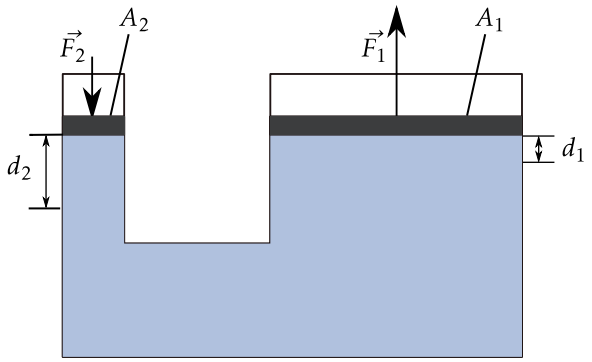
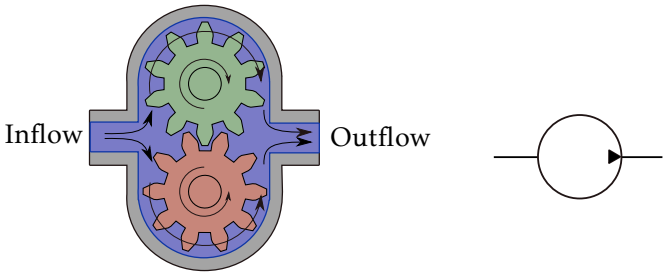


Figure 2.1: Basic principle of hydraulics (based on Figure from [6] p.85). Smaller pressure surfaces result in higher output force at the larger surface.

structurally identical systems can di er based on their physical properties, stemming for example from fabrication variance or defects. This can be used to detect anomalies in a system. To verify that a system's design fulfills requirements even after wear and aging, reliability engineering techniques are used.



(a) Schematic. (b) Symbol.

Figure 2.2: Schematic of a gear pump and symbol.

2.1.2 Reliability Testing

Reliability engineering is the discipline of estimating the lifetime of physical systems and its sub-components. Reliability testing is a valuable tool in hardware development to build long-lasting systems. The IEEE Standard computer dictionary defines reliability as "the ability of a system or component to perform its required functions under stated conditions for a specified period of time." (cf. [19] p.170). In hardware reliability testing, the goal is to verify mechanically that a product will function as intended during its whole life cycle and despite material aging. Typically, for hardware systems the number of failures over time follows a bathtub curve [11], as shown in Figure 2.4. A certain number of systems fails early due to production errors. Towards the end of the maximum lifetime, more parts fail due to fatigue from stress that results from endured load. Reliability engineering is not only important in safety critical domains. Customer

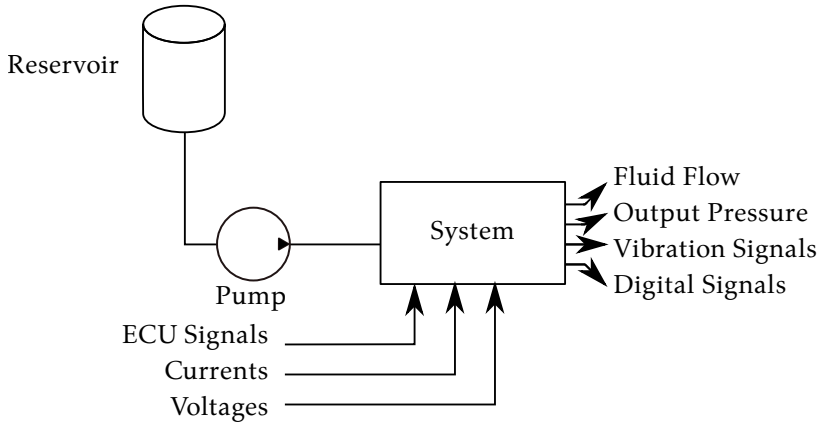


Figure 2.3: Schematic of a test bench for a hydraulic system and recorded sensor signals

satisfaction [20] and ecological sustainability of products also depend on longevity, while over-use of material can waste resources.

To make a statistically sound assumption about the lifetime of a system, accelerated fatigue and strength testing is conducted on a certain number of parts [21]. In these tests, the loads characteristic to the ones expected at operation are applied in the form of e.g., voltages, pressures, and forces. One of the established types of testing are "run to failure tests." Figure 2.5 shows how empirically, the density function of the failure probability can be derived from component failures over time. Based on the test results, the possible lifetime of a part under normal conditions and the average life expectancy of a system can be estimated [20].

In such tests, the inputs of the system are applied as pre-programmed sequences, which are repeated until the objects fail. This means that the measurements can be processed as periodic or seasonal time series. It can be assumed that multiple, structurally identical parts are assessed at the same time, using the same testing program and load distribution. It is of interest to detect anomalies in these tests early, to reduce the time needed for feedback cycles and to support root cause analysis, which is why tools for anomaly detection in measurements from such test benches are the focus of this thesis. Furthermore, since the focus of these tests is the rapid aging of the parts, it is important to also keep the concept drift of the measured data into account which happens due to normal aging in the system.

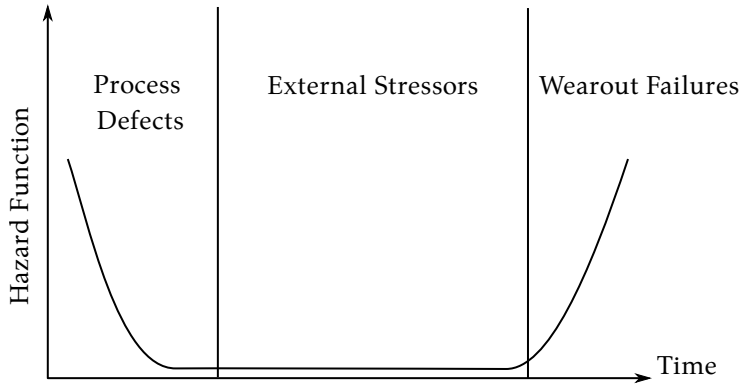


Figure 2.4: "Bathtub" curve for failure probability with the most probable reason of failure. Most failures occur during the beginning and the end of the average lifetime (based on Figure from [21], p. 251)

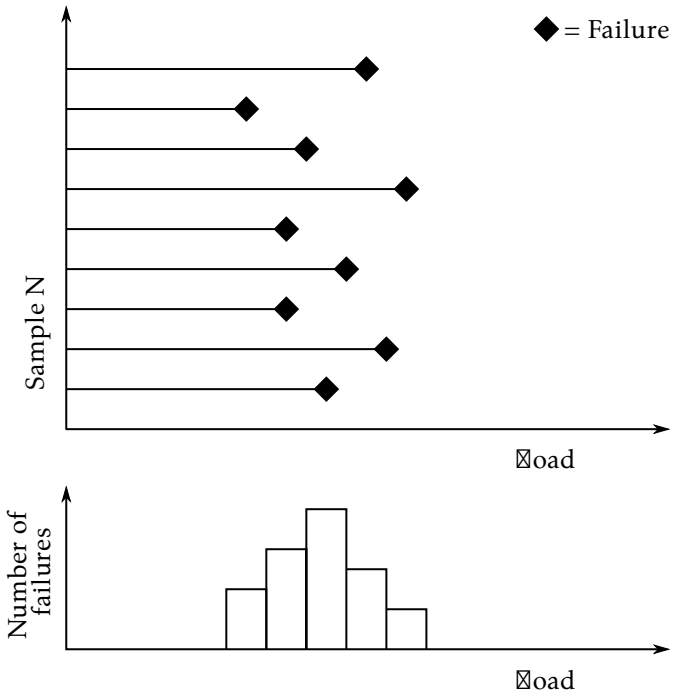


Figure 2.5: Experiment based estimation of failure probability. Based on the histogram of failures, it is possible to estimate a statistical distribution. (based on Figure from [21], p. 252))

2.2 Time Series Foundations

In the following, background information on the analysis of time series is provided, which will be utilized to analyze measurements from hardware.

2.2.1 General Properties

Time series are sequence of values with temporal information. This means that the order of the points is of importance. They have, among others, the following properties [22]:

- Periodicity/Seasonality: in many use cases, time series can be divided into pieces of similar length and similar shape, for example with yearly weather temperature data. Comparison between repetitions then brings additional information
- Univariate/Multivariate: Like images, time series can consist of one (uni) or multiple signal channels
- Trend: the average value of the time series
- Sampling time: the temporal distance between the points in the time series.
- Features in time or frequency domain: depending on the use case, relevant features can be visible more the time domain (i.e., measurement as-is) or using spectral analysis to detect periodic features e.g., by using the Fourier-transform (shown in Section 2.2.2).

During hardware testing, it can be assumed that the recorded data is multivariate since multiple signals are recorded at the same time. Additionally, the testing steps are pre-programmed, which means that some methods of seasonal time series analysis can be used. Due to aging in the systems, changes in trend are probable. Finally, the data is recorded with constant sampling rates. Both features in the time-domain as well as the frequency domain will be investigated in this thesis.

2.2.2 Preprocessing Methods

Based on a measurement time series, several kinds of preprocessing are available, used for e.g., noise reduction or feature extraction. Examples for de-noising methods used in this thesis are mean and median filtering. These filters are implemented by using discrete filter masks of a certain size. This is demonstrated in Figure 2.6 for the mean filter of a size 3. The filter iterates over the input sequence and produces an output value for each value. For the mean filter, this output consists of a sum of the product of the filter's values with the input pixels at the given position. For a median filter, the output consists of the median of the input values at the filters position.

The di erence in output between the two is demonstrated with an example signal in Figure 2.7. The original input shown in Figure 2.7a has an outlier value at $Time = 5$. Using a mean filter, this outlier as well as

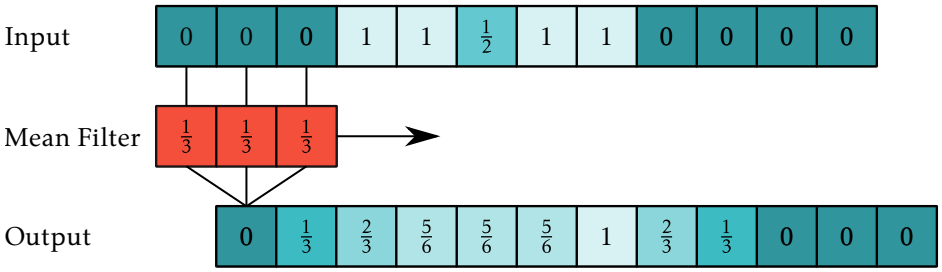


Figure 2.6: Mean filtering using a 3 by 3 filter kernel with values of $\frac{1}{3}$ each.

the boarders of the signal at $Time = 3$ and $Time = 9$ are smoothed. Using the median filter, however, only smoothed over the outlier value and the boarders remain intact (Figure 2.7c).

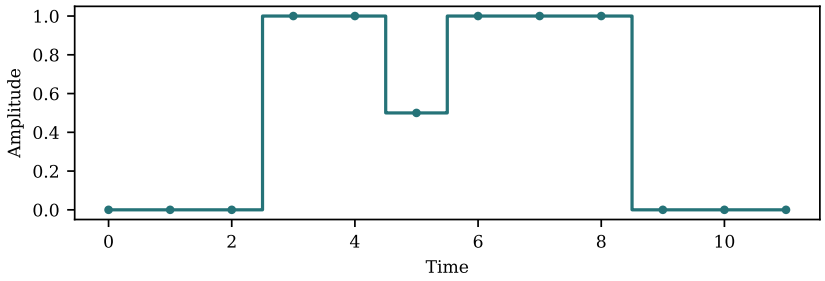
As shown, time series data can be examined in the time domain, i.e., as an amplitude of a value e.g., a signal) over time. Time series also o er the possibility to be analyzed in the frequency domain, which enables the analysis of other properties of a system. An example for this is 50 Hz noise in currents and voltages caused in electric systems due the voltage frequency in the main grid. This can be identified and mitigated by processing data in the frequency domain. Time series can be transformed to the frequency domain by using e.g., the Fourier transform, or the Wavelet transform. The Fourier transform (FT) decomposes a given signal into sine- and cosine functions of di erent amplitude and phase (shown in Figure 2.8). This results in the frequency decomposition of the complete measurement, which means that isolating certain events to a point in time is not possible. To increase the spatial resolution in the frequency domain for longer time series, short-time Fourier transform (STFT) can be applied instead. This means that the signal is subdivided into smaller sections before applying the FT. The size of the segments influences the temporal and frequency resolution of the output. This reduces the maximum number of frequencies that can be examined due to the Shannon-Nyquist sampling theorem. Therefore, with smaller time window size it becomes more visible when certain events occur in the signal, but the frequency resolution decreases.

Frequency domain analysis is useful for domains with oscillating signals, stemming from e.g., electrical alternating currents or mechanical vibration, as shown in Chapter 6 and 7.

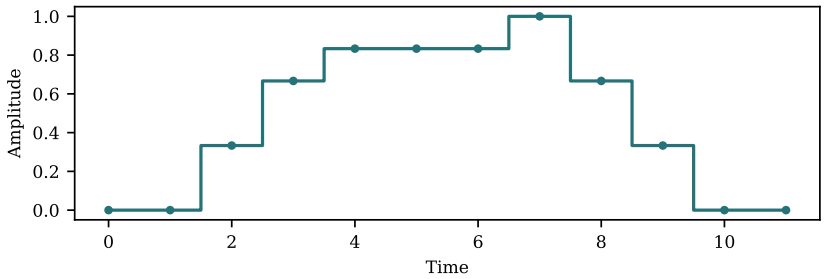
2.2.3 Anomaly Types

In the context of this work, the test bench in question produces various pre-programmed maneuver measurements, which are saved on a server already separated along repetition lines and labeled into classes for different repetition shapes. This results in large amounts of time series data. Anomaly detection for such data is a research field of high interest.

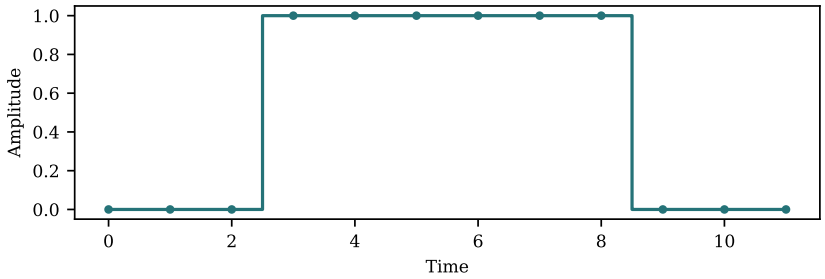
Diverse types of time series anomalies are defined as part of state of the art [23]–[25]. In their review, Chandola et al. define three main types for data in the time domain [25]and demonstrated in Figure 2.9:



(a) Original signal.

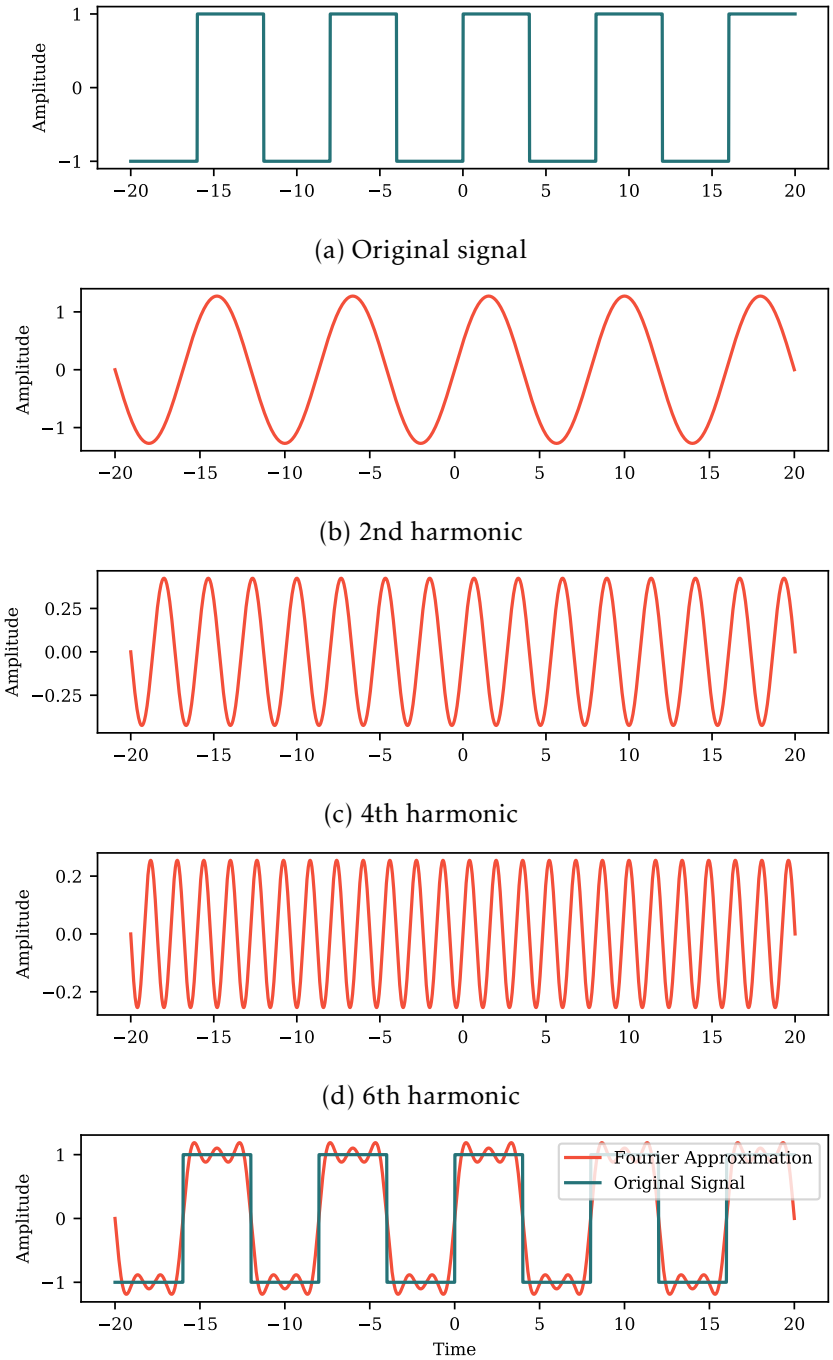


(b) Signal after mean filtering with a kernel size of 3.



(c) Signal after median filtering with a kernel size of 3

Figure 2.7: Comparison of mean and median filtering on a time series input signal.



(a) Original signal

(b) 2nd harmonic

(c) 4th harmonic

(d) 6th harmonic

(e) Sum of 2nd to 6th harmonics versus original signal

Figure 2.8: Fourier decomposition of a rectangular signal. 1st, 3rd, 5th ... harmonics are zero.

- Point anomalies are data points that are di er strongly from the rest of the data (Figure 2.9a).
- Contextual anomalies are values which occur as normal in other parts of the time series but are anomalous due to other values in their context/neighborhood (Figure 2.9b).
- Collective or pattern anomalies are a group of points is anomalous compared to the rest of the dataset (Figure 2.9c).
- Change points are anomalies where there are changes in the time series, such as a change in frequency, incline, mean or standard deviation (Figure 2.9d).

Additionally, there are anomalies which detectable primarily in the frequency domain, particularly for stationary, oscillating time series. This is illustrated in Figure 2.10. Here, both datasets are sinusoidal data with superimposed noise. For the normal case, the noise is normally distributed. In the abnormal data, the noise contains additional sinusoidal oscillations, which appear as additional peaks in the frequency spectrum of the signal.

For this reason, throughout this thesis, time series anomalies in the time and the frequency domain will be examined.

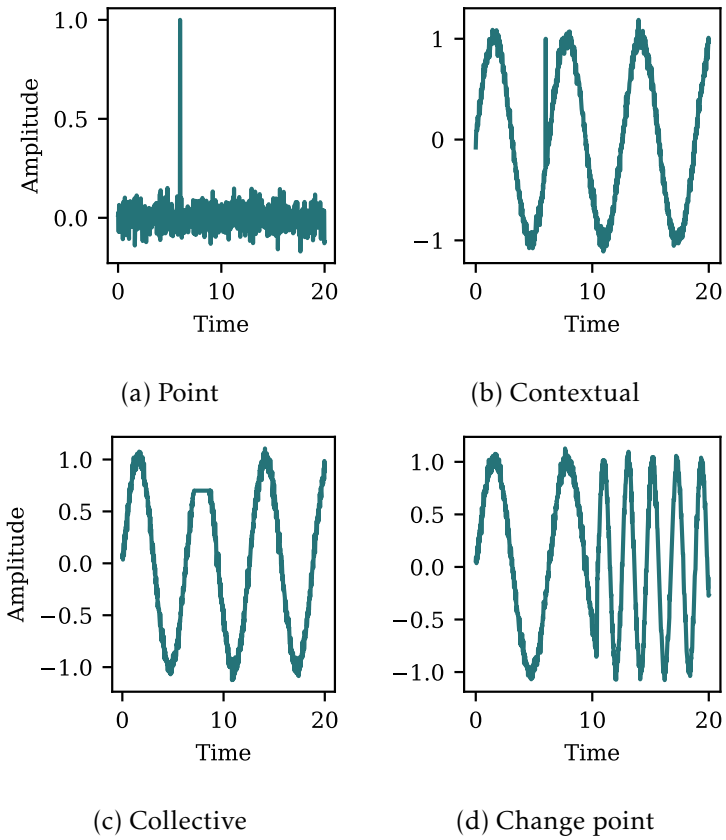
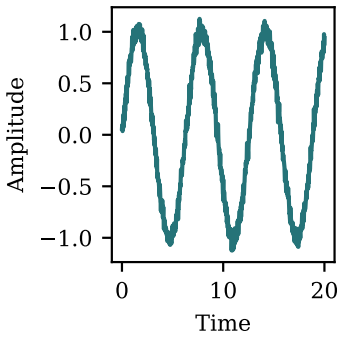
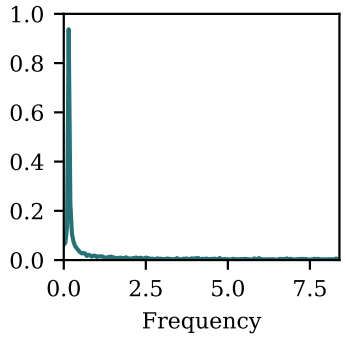


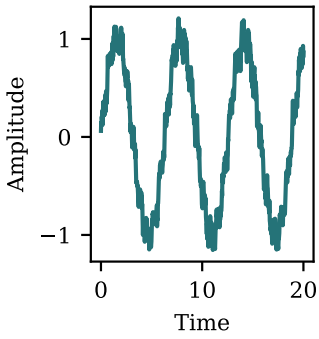
Figure 2.9: Examples of anomaly types in the time domain ((a) and (b) based on Figures from [23], p. 6482).



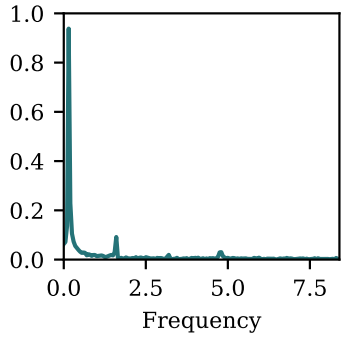
(a) Original signal (normal)



(b) Frequency representation of (a)



(c) Original signal (abnormal)



(d) Frequency representation of (c)

Figure 2.10: Examples of anomaly types in the frequency domain.

2.3 Related Work on Anomaly Detection for Physical Systems

This section will discuss at several aspects of anomaly detection (AD). Anomaly detection is a common application area of data science methods and is of interest for various domains, e.g., medicine, finance and fraud detection, industrial applications and mechanical units and Information Technology (IT) [25]. In recent years, the domains of Cyber Physical Systems (CPS) and Internet of Things (IoT) and PHM (prognostics and health management) have also gained importance (see surveys [23], [24]) . Subjects of anomaly detection in physical systems are e.g., gear boxes [26], wind turbines [27]–[29], hydraulics (with valves, pumps, gear, shaft) [30]–[33], server infrastructure [34]–[36], electrical motors [37], turbofans[38], vacuum pumps ([39]), and aircrafts [31].

2.3.1 Data Visualization and Exploration

The first aspect of this thesis is the visualization of time series of the test benches. Since the datasets in most parts will be unlabeled, the first step should be the understanding of data, ideally using visualization [40]. This is the goal of research areas such as Knowledge Discovery and Data Mining (KDDM) . KDDM is the act of extracting new knowledge from a data base [41]. KDDM is defined as several steps, with the complete pipeline including deployment of the application and database.[42]. [41] present a model for KDDM process with the steps "Domain understanding, data preparation, data mining and evaluation" of data ([41], p.). Data Mining (DM) is defined as methods to enable users to analyze data and enable pattern recognition for data. A key step in data mining is the visualization of data along with precomputed features, to be able to estimate which tools for anomaly detection are appropriate. One way to distinguish AD methods is into supervised and unsupervised approaches.

2.3.2 Supervised vs. Unsupervised Anomaly Detection

Supervised anomaly detection can be applied when the given training data is fully labeled and both normal and abnormally labeled data points are given [43]. Conversely, unsupervised anomaly detection methods can be applied if only unlabeled data is available. Given only data labeled as normal, semi supervised anomaly detection methods can be applied. This means that the algorithm computes a decision boundary based on the normal samples, and new data points outside of this boundary are classified as abnormal [44]. The three types are shown in Figure 2.11. Examples for supervised methods are [45] for distance-based anomaly detection for circuits using a threshold, [32] for supervised anomaly classification for hydraulics using various feature reduction and classifiers. Unsupervised methods are, e.g., [46] with model-based anomaly detection for general time series, [47] for device failure detection and [34] for server access count anomaly detection.

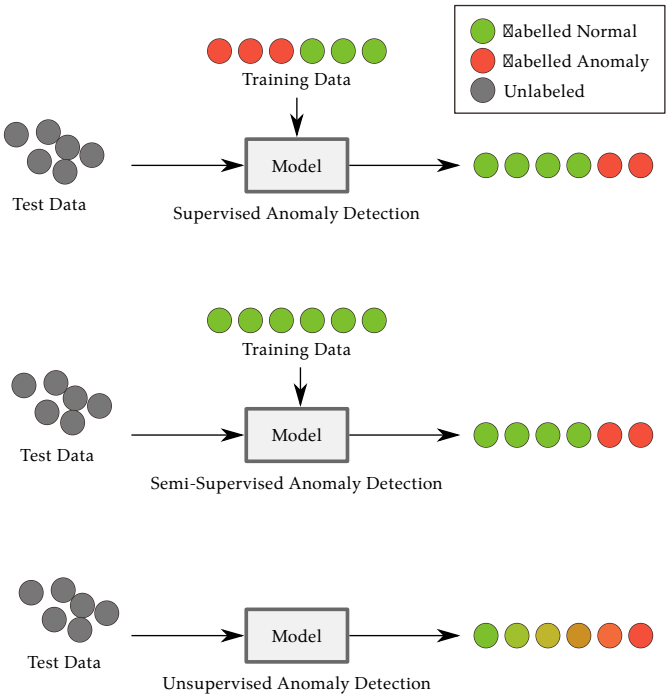


Figure 2.11: Di erent strategies of anomaly detection, depending on the training data available (based on Figure 1 in [44])

2.3.3 Model vs. Data Based Features

In the context of this thesis, anomaly detection methods are further classified into data-based or model-based approaches. In the following, a model-based approach means that a model is used to predict the "normal" behavior of a time series. In case of time series from physical measurements, this overlaps into the domain if modeling a system using a digital twin model, where the model predicts the output of a system based on measured inputs [48]. The di erence between predicted and actual measurement can then be used to detect anomalous behavior. Examples for model-based methods can be numerical simulations using state space models or models derived from data using machine learning, such as recurrent [49] neural networks [46] or neural ordinary di erential equations [50]) . Other examples are Hidden Markov Models, auto regressive models or Kalman filters [51]. But if the dynamics of a system are too complex or unknown, data-based methods should be preferred [49].

Data based anomaly detection describes the classification of data, without the prior training of a predictor for the ideal behavior. Di erent algorithms for this are part of the state of the art. Statistical methods have been used for bearing fault detection [52] and for server access count anomaly detection[34]. Helwig et al, for example, perform various feature engineering and reduction methods on time series data, before applying supervised anomaly detectors in the shape of Linear Discriminant Analysis, DNNs and Support Vector Machines (SVM) [8]. In IoT, neural networks are common models for Industrial use cases, not so much simpler models like decision trees and regression models [24]. Other commonly used models,

are Support Vector Machines (SVM) [8], deep neural networks (DNN), and k-nearest-neighbor (KNN) or Local Outlier Factor (LOF) [53] can be applied [27] [54].

2.4 Chapter Summary

This thesis focuses on measurements in the shape of repeating time series maneuvers. Time series analysis in general is a large and versatile research domain, which deals among other topics with time series visualization, analysis and anomaly detection. The goal of this thesis is to cover diverse aspects of this are with the focus on the specialized use case of hydraulic test benches, in order find methods that benefit technicians and engineers. For this reason, Chapter 3 will first investigate visualization strategies of periodic time series. Based on this, Chapter 4 will examine model-based anomaly detection for hydraulic systems using recurrent neural networks. This approach posed unexpected challenges, which is why in instead in Chapter 5 a data-based, unsupervised method is developed. Di erent preprocessing steps are compared in this process, and the proposed algorithm was shown to be ine ective for the given data set for features in the frequency domain. Since it is known that time series in the frequency domain is e ective for vibration-based anomaly detection, in Chapter 6, anomaly detection for frequency domain data is examined in depth using convolutional neural networks. Since the latter are black box models, the final Chapter of this thesis develops a new XAI method for explaining classifier models for data in the frequency domain. To conclude this introduction, the main five Chapters in the thesis have the following focus points:

- Data visualization (3, 5, 7)
- Supervised (6) vs. unsupervised (5)
- Data-based(6, 5) vs. model-based (4)
- Time domain (3, 4, 5) vs. frequency domain anomalies (5, 6)

3 Visualization of Periodic Time Series*

This thesis examines the possible uses and benefits of data science methods in the domain of anomaly detection in the context of hardware test benches. An important part of any data science research is data visualization, i.e., a graphical representation of data to retrieve added knowledge and insights on the data visually. This is a useful and necessary step before implementing and training an AI algorithm. But also at prediction time, human users need to interpret a model's predictions efficiently and precisely. Because of human pattern recognition abilities, in some cases right visualization can even replace the need of an AI model, if a user can come to a decision solely based on shown information.

Periodic (or seasonal) time series are sequential data sets of repeating, similar patterns, which can have changes in trend and periodicity. Since the measurements from test benches is in the shape of repeated, preprogrammed signals, they can be interpreted as periodic time series. Possible problem types include anomaly detection, system state classification, change point detection, or also trend evaluation. A challenge of long, periodic time series are the substantial amounts of data, which stem from high sampling rates and/or long recording times. Figure 3.1a shows a plot of periodic data for univariate anomaly detection of access counts of a server. The data is shown as a function of a metric over ca. nine days.

Even though this is a common way of visualizing such data sets, it is often inefficient. To evaluate the data points that were classified as anomalous by an AI, a human would have to visually compare multiple data points at the same position in a period over many pattern repetitions. This process of assessing the data, either for manual detection of anomalies or for checking the correctness of an AI algorithm is therefore time intensive and prone to error. This poses a problem not only when labeling data, but also when trying to explain an algorithm's prediction based on an input. Seasonal plots, for example, display data split across season lines and plotted over the duration of one season. While better than the prior described visualization, in case of varying periodicity or a trend in the data, it is still sub-optimal. Therefore, this chapter examines possibilities for pre-processing, visualization, and interactivity for the analysis of seasonal data in the univariate and multivariate case. It proposes to plot seasonal data over the length of one season by splitting the data at season lines (Figure 3.1b to Figure 3.1c) and mitigating changes in trend and periodicity

*Parts of this chapter have already been published in "D. Neufeld, "Visualization Methods for Periodic Time Series Data", in *Lernen. Wissen. Daten. Analysen. (LWDA'21)*, (Munich, Germany), Online: CEUR Workshop Proceedings, Sep. 2021, pp. 1–10. [Online]. Available: https://ceur-ws.org/Vol-2993/paper-16.pdf"

by applying pre-processing steps, as shown in Figure 3.1d. Combined with this, steps for interactivity of the charts are proposed. The methods are adaptable towards the specific use case and domain of new problem statements. The core concepts are shown based on three examples: an ECG signal, the server access count at Twitter and a data set for condition monitoring of a hydraulic system. Additionally, a user interview with test bench engineers is conducted to quantify the benefits of a prototype implemented with this method.

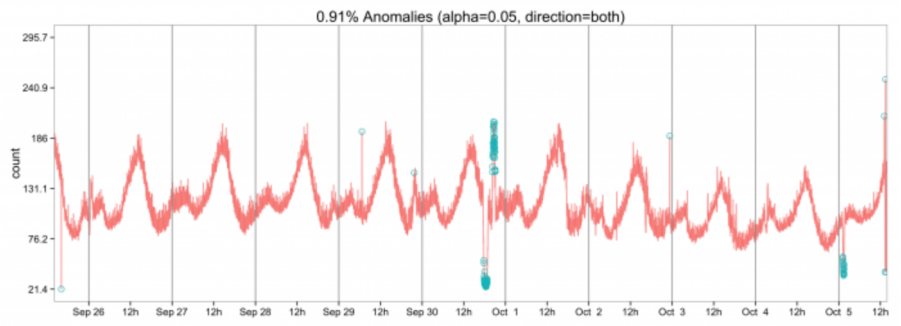
The methods shown are presented as a toolkit for the visualization of periodic data. Expert domain knowledge is still needed to select the useful ones for a new use case and to judge the results. Furthermore, with big data applications, further numerical optimizations can be necessary, like down sampling, pre-clustering of data and data reduction. Otherwise, the performance of the visualization can su er, resulting in worse user experience.

First, this chapter shows related work examples on anomaly detection and visualization for periodic data and interactivity methods. Then the components of the presented method are explained. Finally, it is demonstrated based on example datasets and a user interview is conducted.

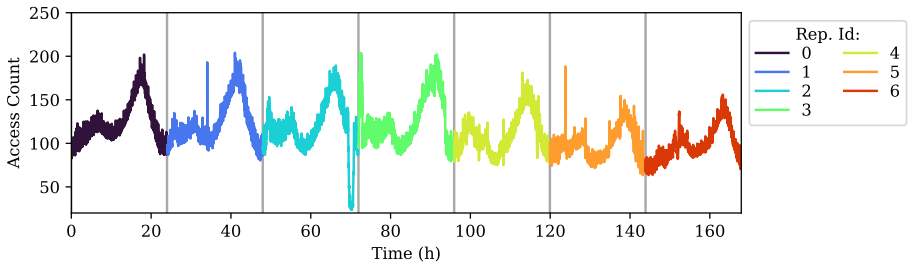
3.1 Related Work

Time series are subject of various kinds of visualization [56]. Periodic time series are a common data type in data science and AI. The proposed visualization approach is applied to three di erent data sets. First is an ECG data set from the Scipy Package [57], a problem type which was evaluated for anomaly detection amongst others by Chakraborty et al.[58]. They extracted the normal signal from multiple signal periods and used it as a benchmark to compare new measurements. The di erence is then classified based on a pre-set threshold. Second is a data set of server access counts over time at Twitter, published by Kejariwal and used by Hochenbaum et al. ([55], [34]). For point- and global anomaly detection of this data set, they first extract the trend and the seasonal part of the data and classify the residual with an Extreme Studentized Deviate (ESD) test. The third use case is a data set for condition monitoring of a hydraulic system with multiple deteriorating sub-components by Helwig et al. (used in [8], [32] and [59] in the domain of supervised state classification. While this data set is aligned in amplitude and time direction, it is a multivariate data set, which adds complexity. This shows that the proposed method can be of interest for engineering use cases as well.

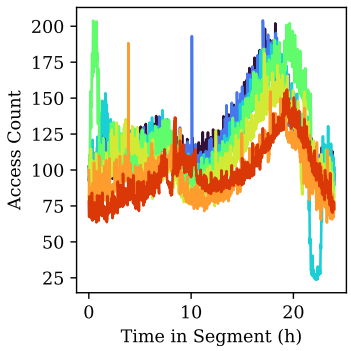
Visualization is a crucial step in data science and can reveal patterns in a data set. Wu and Keogh [60] reviewed the progress in machine learning for time series anomaly detection in the last years and discovered that many of the newly published pieces work use complex models (Deep Neural Networks) for "trivial" problems that "can be solved with a single line of standard library MATLAB code" (cf. [60], p. 2). They have shown that most of these published pieces of work have been applied to trivial problems. Many of the data sets examined in their work are periodic time series.



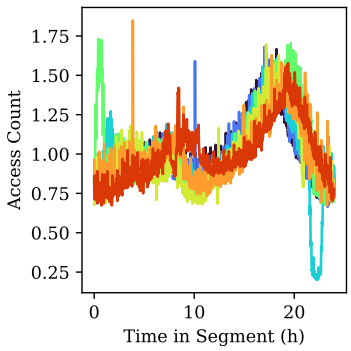
(a) Visualization of anomalies the data of [34] as a red line plot over the range of nine days. The blue dots show the anomalies computed by an anomaly detection algorithm (c.f. [55]).



(b) Server access count over time. Vertical lines are plotted between period repetitions of 24 hours.



(c) Per-day data plotted above each other.



(d) Data de-trended and fitted over time.

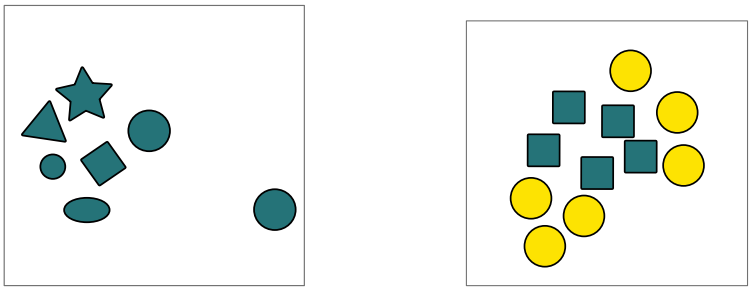
Figure 3.1: Plots of data that was published with Twitter's SESD algorithm [34] for the anomaly detection in server access counts. Figure 3.1a is from the publication, Figures 3.1b through 3.1d are based on the proposed method.

Thorough visual inspection of data can help in designing more efficient machine learning models and in avoiding over-engineering. Other pieces of related work examine the visualization of periodic time series data. In their survey, Fang et al. [61] discussed visualization methods for multivariate time series. They show an approach called the calendar view, a plot for seasonal data. While similar to the proposed method in this chapter, each season of the data is plotted in a multi-line plot over one common time axis with the length of the season, for example the data of a year is divided into months and displayed over the range of one month. The plot is combined e.g., with a calendar widget to filter the days to show. Fang et al. cite examples by Wijk and Selow [62], Macas and Machado [63] and Buono and Balducci [64] for this. Matkovic et al. [65] developed a GUI (graphical user interface) application visually similar to the one from this chapter with the focus on simulation result visualization. For this, they plot generated data based on simulation parameters chosen by the user. The resulting data was also plotted as a multi-line plot over one time axis, improving the visibility of di erences in the results, but is missing the alignment steps proposed here. Gjika et al. [66] also examine the visualization of periodic data for the domain of hydropower plants, creating seasonal plots for each channel. The data is split at each repetition and plotted over the duration of one seasonality. The proposed method similarly entails to present time series segments in a multi-line plot over a common time axis, but in contrast to other work, the benefits of de-trending and time o set removal are also shown.

Regarding interactivity of visualization, Lin et al. [67] have presented a graphical user interface (GUI) tool called VizTree, for the visualization of clusters of patterns in large time series databases based on augmenting suffix trees. That publication is focused more on visualizing common sub-sequences across several time series, while this approach is optimized for the special case of periodic time series, instead.

A di erent data science domain, Functional Data Analysis, routinely uses plots visually like the one in this picture. Hyndman and Shang [68] developed rainbow plots for visualizing continuous, smooth functions over an identical data range. This results in multiple line plots where each line is colored with a di erent shade, chosen from a rainbow color map, like the plots in this chapter. The di erence is that the proposed method works with a broader range of application domains, and therefore depending on the use case, also entails di erent pre-processing steps (de-trending, fitting along time axis) and interactivity functionality.

Interactive visualization is a valuable tool, especially for end users. Yi et al. [69], for example, published a review of interactivity categories for data visualization. They named the taxonomic units of Select, Explore, Reconfigure, Encode, Abstract/Elaborate, Filter, and Connect ([69], p. 3). Adding to this, known Gestalt principles (first described by Wertheimer [70]), describe how humans visually interact with data representations and can be used to improve complex visualizations. In the following Chapter, it will be described, how the proposed approach follows interactivity and Gestalt principles.



(a) Law of proximity. Elements drawn closer appear to be of the same group.

(b) Law of similarity. Elements of similar shape and color appear to be of a group.

Figure 3.2: Illustration of two of the Laws of Visualization by Wertheimer [70].

3.2 Methods

Having discussed plotting approaches and different use cases where periodic data is relevant, the proposed visualization methods are described next. To support the visual evaluation by humans, the visualization is designed to compress the plot of a long, periodic time series such that related points across all period repetitions are shown closer together. This is to better use the Gestalt principles of law of proximity (elements that are close together, appear as a unit) and law of similarity (elements that are similar, appear as a unit). The former shows that elements which are closer together appear as a unit, compared to distant objects (Figure 3.2a). The latter means that objects displayed similarly, for example by color or shape, appear to belong to the same group (see Figure 3.2b).

The goal is to enable the visual estimation of the data distribution over the duration of one period. This improves the visibility of anomalies, since they are data points that have above average distance from the rest. The proposed approach consists of the following steps:

1. Split the complete time series along period borders
2. (Optional) Remove the time offset using correlation (used in Section 3.3.1 and 3.3.2)
3. (Optional) Remove the trend of the repetitions (used in Section 3.3.2)
4. Plot line segments over one common time axis
5. (Optional) Enable interactive highlighting (used in Section 3.3.3)
6. (Optional) Create other plots, like an anomaly metric in the multivariate case (used in Section 3.3.3)

The computation of the periodicity of the signal is use case dependent. For example, in an ECG case, the period of the signal can be found by counting the peaks in the signal above a certain threshold (see Figure 3.3d). Splitting the signal across period lines is the first and main aspect of

the presented work, and if there are no other o sets or noise present, this is the only preprocessing step before plotting. Depending on the problem domain, the time and trend o sets between the signal repetitions can be removed to improve visualization. The computation of the trend component in the examples is done based on the seasonal trend decomposition (STD) of the time series, shown in Listing 3.1. Depending on the data set, one can choose a multiplicative or additive removal of the trend. Then, the time o set (in x-axis direction) of the segments can be removed using cross-correlation. For this, the version implemented in the Scipy library [57] signal.correlate as described in Listing 3.2. For two sequences with the length N and M, correlate has the time complexity of $O(NM)$. For the examples it was chosen over a Dynamic Time Warping algorithm, which in its basic implementation also runs in $O(NM)$, but in practice runs much slower. The data segments are then ready for visualization.

The shown examples were implemented and plotted in Python 3.7 using Matplotlib.pyplot [71]. The color map used is the standard "turbo" color map, which is a continuous over a wide array of colors. In the multivariate case, each channel of the time series is plotted in a separate sub-plot. Since lines can cover each other in the plot depending on painting order, it is useful to provide the possibility to highlight and hide certain lines in the plots on mouse-click. This was implemented using the mplcursors library [72]. For the multivariate case, highlighting all channels of a repetition after mouse click and displaying the repetition index on click is a useful feature. Just by implementing the proposed visualization algorithm with Matplotlib (a Python plotting framework), it is possible to cover many interactive features described by Yi et al. [69]: Select (by highlighting of lines on mouse click), Encode (by using additional sub-plots on the data), Filter (by enabling the hiding of lines on click) and Connect (by enabling that all relevant lines of a repetition are highlighted on mouse click).

```
import numpy as np
from statsmodels.tsa._stl import STL

def remove_signal_trend(series, repetition_count):
stl = STL(series, period=(len(series)) // repetition_count)
res = stl.fit()
return series - res.trend
```

Listing 3.1: Example method for removing the trend in time series.

```
def fit_signal_time(wanted_signal, other_signal):
sampling_time = wanted_signal.sampling_time
shift_index = scipy.correlate(other_signal.y, wanted_signal.y).
   argmax()
if shift_index != 0: # shift_index is 0 if there is no
   correlation
shift_time = wanted_signal.time[-1] - other_signal.time[0] -\
shift_index * sampling_time
other_signal[signal_name].time = other_signal[signal_name].time
   + shift_time
```

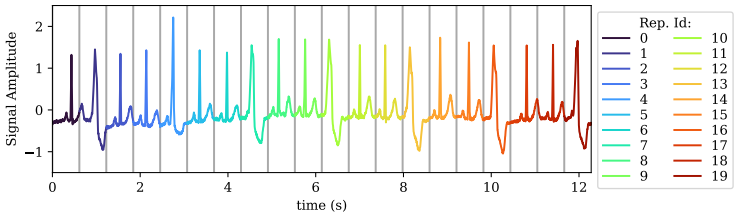
Listing 3.2: Code for adjusting the time axis of one signal (other_signal) to fit the signal to another (wanted_signal) signal.

3.3 Examples

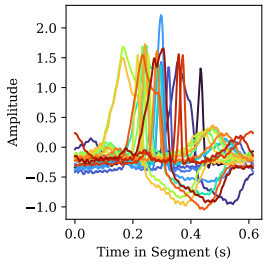
The presented methods are shown on three di erent data sets: the ECG data set from the Scipy package [57], the data set published by [34] in the presentation of Twitter's Seasonal ESD algorithm, and the hydraulic systems condition monitoring data set by Helwig et al.[8].

3.3.1 ECG Evaluation

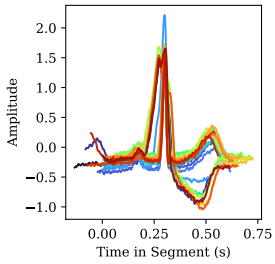
For demonstration purposes, a section of the data set was chosen that has a constant periodicity and no trend (Figure 3.3a). Just splitting the time series into its number of periods yields Figure 3.3b, which is a very noisy plot due to slights o sets in periodicity in the signal. Applying Listing 3.2 to the split data results in the data set shown in Figure 3.3c. Plotting this side by side with an standard ECG graph (adapted from [73])in Figure 3.3d, it becomes obvious even to laypeople that the signal often displays an anomaly in the ST part of the signal, and the R peak is often broader and less pointy than normal. Without this plot, a human would have to compare all distinct parts of the signal along all repetitions of the period. This type of plot can also be used for de-noising. If all the lines are plotted with higher transparency (an alpha < 1.0), noisy parts of the data are displayed less prominently.



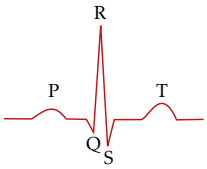
(a) Signal over time.



(b) Split along period lines.



(c) Fitted along time axis.



(d) Normal ECG Signal.

Figure 3.3: Application of proposed visualization to part of Scipy's ECG data set.

3.3.2 Twitter Server Access Counts

In the use case of server access count anomaly detection, Kejariwal shows an anomaly detection algorithm which is based on Seasonal Trend Decom-

position [34]. Figure 3.1a shows a visualization from the blog post. As can be seen, the data set shows a slight downwards trend over the span of the data set. Based on this plot it is difficult to judge the performance of the algorithm closely: focusing on the plot from the evening September 29th, an anomaly was detected, but it is difficult to interpret why this was an anomaly, when the peak in the mornings of October 3rd and 4th were not. In terms of visualization, de-trending can be applied because the use case is focused on point anomalies. Therefore, the trend removal (Listing 3.1) and amplitude shift removal (Listing 3.2) were used. The resulting plot in Figure 3.1d visualizes the processed data set. Anomalous peaks are now visible. The day of the anomaly can be displayed, again, by selection and highlighting via mouse-click.

3.3.3 Hydraulic Anomaly Classification

The data set for condition monitoring in hydraulic systems by Helwig et al. [8] is a multivariate data set. It is designed for supervised anomaly classification of a hydraulic system that has several sub-components which can fail. The data set is recorded on a hardware test bench, which means the measurement cycles of the systems are repeated as uniformly as possible. This is a typical use case in hardware engineering, especially in the domain of reliability analysis (see e.g., Birolini [74]), where load is applied to multiple structurally identical hardware systems over a long amount of time to estimate the systems behavior over its product life cycle. The time series run for 6 seconds, with a maximum sampling rate of 1 kHz. Among the recorded signals of the hydraulic system are six pressure (P1-P6) and temperature (TS1-TS4) signals. For the plot in Figure 3.4, 10 normal cycles were chosen (index 0 through 9) and two abnormal cycles (Id. 10 and 11), where the cooling sub-component was damaged. The removal of trend and time o set is counter-productive because the trend over time is an important sign of anomaly. All channels were plotted in separate sub-plots. Additionally, an anomaly metric was computed by taking the median of all measurements of the normal and calculating the Mean Absolute Error (MAE) of each measurement repetition towards it (further described in Chapter 5). This is visualized in the "MAE across channels" sub-plot: it displays the distance of each repetition per channel normalized to $[0, 1]$ in a parallel coordinate plot. This way, it is visible that 10 and 11 have unusual high deviation from the other cycles, and it is visible which channels show the most deviation. This can be used by a human to decide, which channel's sub-plot should be looked at closely. Due to its multivariate nature and complexity of data, Figure 3.4 shows the usage of an interactive highlighting functionality. After clicking on one line in this example the line 11 in the MAE plot, all corresponding lines are highlighted in black in all sub-plots. Since the plot was implemented in Matplotlib, it is also possible to interactively zoom in on each plot if required.

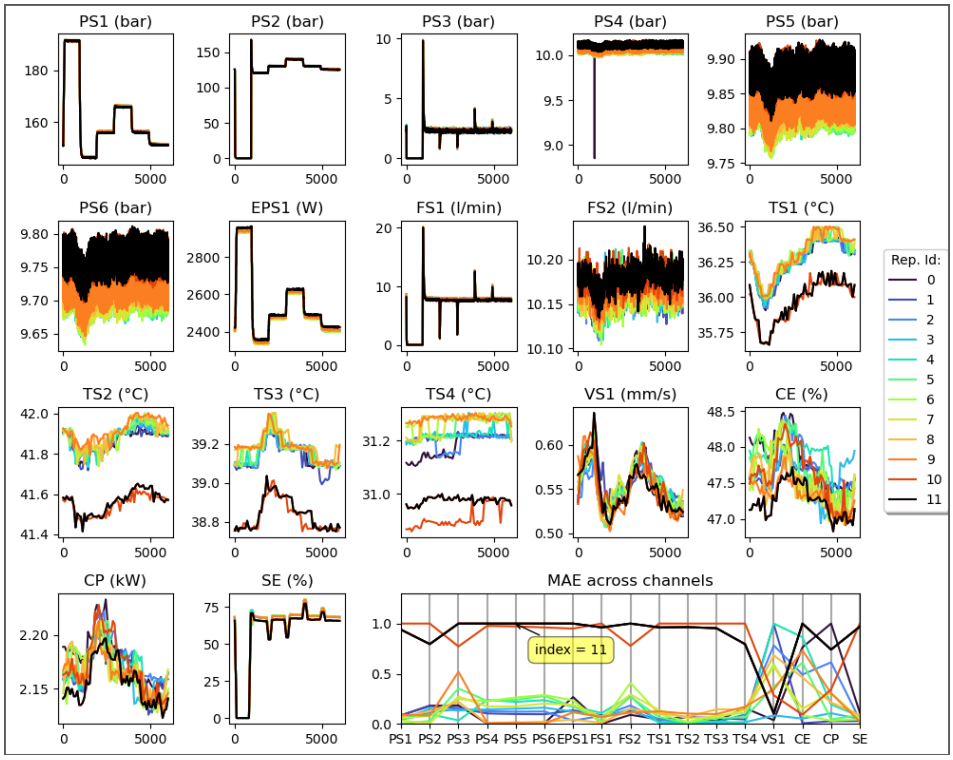


Figure 3.4: Visualization of the multivariate data set from Helwig et al. [8]. Repetitions 10 and 11 were chosen from a run with a broken sub-component. Clicking on one line in the plot highlights all other lines belonging to the same repetition. The plot titled "MAE across channels" shows a per-channel distance metric, where each line belongs to one repetition.

3.3.4 Test Bench Anomaly Detection

The method was also conducted with engineers for one use case, where twenty measurement repeats with eight signals had to be evaluated for two hundred types of measurements, while also comparing the data to a specification given in a csv file. Originally, this data was visualized as one time series plot per signal type. The engineers' task was to find anomalous measurements out of the twenty. The data for this cannot be shown, but the new visualization was similar to the one shown in the prior Section. After using the visualization, two reliability engineers were interviewed on the time savings using the new plots. The new approach had an estimated benefit of 80 hours compared to the old one (see Table 3.1). They estimated that the new visualization leads to 80% of review time reduction and better review coverage.

3.4 Chapter Summary

In this chapter, the visualization aspect of data science was investigated. A collection of visualization method for periodic time series data was presented, which were shown to be flexible and adaptable to other domains.

Table 3.1: User feedback on the benefit of the proposed approach for one scenario of test bench data.

	Conventional	Using the Prototype
Time per repetition	5 min	1 min
Repetitions per test run	200	200
Runs per year	6	6
Time cost	100h	20h
Accuracy	Random samples	All repetitions at once Repetition accuracy visible Relative comparability

Demonstrations were provided for three different types of data, and an expert interview was conducted to give a first impression on the benefits of the method in the context of test benches.

Still, as usual with data science, the choice of optimal visualization methods depends on the problem domain. For example, for the domain of hydraulic system supervision, it would be incorrect to remove the trend of the data, because it can indicate when a part starts to fail.

There are opportunities for future work regarding user interfaces for time series data interaction: One is the benefit analysis of the method for the labeling of data for training, with regards to accuracy and time savings. For this, the user interface must be combined with a data base for keeping track of the labeled data points. Additionally, to prepare the method for actual big-data approaches, it can be beneficial to research the clustering of multi-dimensional time series. This could provide another layer of information the data.

Having shown possibilities to visualize the data that is the focus of this thesis, now the focus will turn to the actual anomaly detection algorithms. Therefore, in the next chapter, model-based anomaly detection for test bench data using recurrent neural networks will be investigated.

4 Challenges in Model-based Anomaly Detection with RNNs

The last chapter showed how to visualize time series data relevant to the use case of hydraulic testing. The remainder of this work examines methods for the anomaly detection in time series from test benches, with focus on methods with results understandable to experts. First, model-based anomaly detection using deep neural networks (DNNs) is evaluated. Unexpected challenges occurred during this research, which are described and the root cause of which are analyzed.

The motivation of this chapter is the search for a data driven model to reproduce the behavior of a physical system based on its measurement data. The resulting model should function as a Digital Twin for use in anomaly detection. Digital Twins are a field of research especially relevant for industrial applications [48]. They are used to model relevant properties and behavior of a system, be it physical or digital, as closely as possible. Digital twins can be used for anomaly detection, for example to model the normal behavior of the system as a baseline, and then classify the di erence based on a threshold, as illustrated in Figure 4.1. Other uses for Digital Twins besides anomaly detection are predicting future behavior and properties, such as the systems reactions to new inputs, and the use as sub models in simulations for the development of larger systems [75].

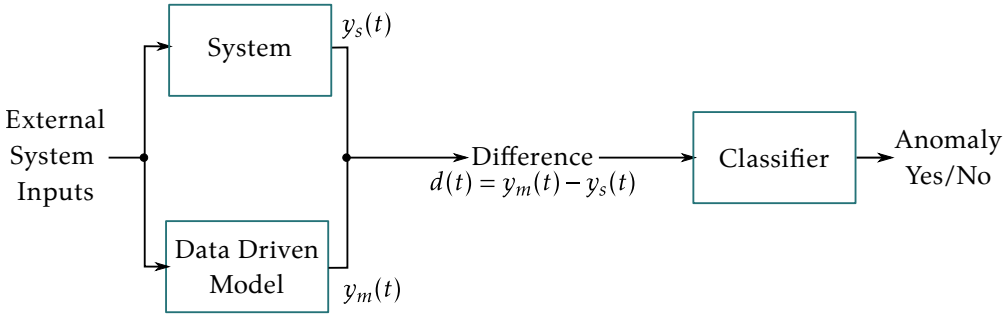


Figure 4.1: Concept of a Digital Twin for anomaly detection: the twin is a model of the system's normal behavior. The di erence between the model and the actual system output is used to detect anomalies using e.g., threshold-based classification.

While there are benefits to modeling a system based on its output data, impediments also exist with a model-based approach. Modeling of physical systems can be a challenging task. Especially if a system changes over time (i.e., there is a concept drift in the data), the model must incorporate this, otherwise it will become inaccurate. In the context of test benches, where parts are damaged on purpose, changes like this can occur rapidly.

In this case, methods to combat this such as model updates must be applied [76]. This leads to the problem of verifying a model's accuracy in terms of anomaly detection. If the discrepancy between a model and the system is too large, this can either be due to inputs that are out of the distribution of the training data, e.g., due to system changes, or because there actually is an anomaly present. Another problem is over-fitting, meaning that the model is memorizing the system's output instead of understanding and reproducing its behavior. Test bench data, i.e., pre-programmed, repeated hardware measurements, are more likely to cause this. Overfitting might be avoided by using cross-validation and regularization methods [77]. Additionally, in practice, the recording of data is an iterative process. Often not all sensors are calibrated perfectly from the start, which can lead for example to wrong encodings, sensor o sets and missing data [78]. In this chapter, physical processes are in the focus, such as changes in pressure and motor currents. The data was viewed prior to training to ensure the correct sensor calibration.

Still, should there be a fast and reliable method to train a Digital twin model based on measurement data, it would be beneficial not only in terms of anomaly detection, but also when simulating the system's responses to novel inputs.

The focus of the following chapter is the modeling of one behavior of a hydraulic system using recurrent neural networks, with the specific functionality of performing an integral of a measure over time being the focus. This means that based on an input $x(k)$ over a time series with the discrete time steps k , the system and its model should be able to reproduce the output $y(k)$ with

$$y(k) = y(k-1) + K \cdot x(k), \tag{4.1}$$

which is the summation of the values of $x(t)$ over time, scaled by a parameter K. A real-world example of this is the filling of a reservoir with the volume flow of a faucet. The volume in the reservoir corresponds to the integral of the volume flow over time. It was found that three state of the art recurrent neural networks fail when modeling this kind of system for long sequences. The goal of this chapter is to investigate the root cause of these issues.

The recurrent networks that are examined are Recurrent Neural Network- (RNN, Jordan [79], Elman [80]), Long-Short-Term-Memory- (LSTM, Hochreiter and Schmidhuber [81]) and Gated Recurrent Unit-layers (GRU, Cho et al. [82]). Recurrent networks have shown great capabilities on text processing [82] and time series and sequence prediction problems of di erent domains such as water level prediction [83], bacterial growth prediction [84] and music and speech modeling [85].

During the research for deep learning-based modeling of hydraulic systems, behavior occurred in recurrent networks that was not yet described in related work: A simple measurement signal at a high sampling rate, which is based on integrating an input, could not be replicated using the recurrent neural network types RNN, LSTM and GRU.

The topic of modeling physical systems is type of dynamical systems modeling. Dynamical systems modeling originally belonged to the domain of mathematics and physics. Nowadays, it is used widely in other areas, such as economics[86], bio medicine [87] and engineering , e.g., as part of control theory [88]. The topic as a whole exceeds the scope of this thesis, which is why this chapter focuses on one objectively simple kind of system (described in Equation 4.1). The models are implemented in the discrete time domain (as opposed to e.g., frequency domain). As recurrent layer types, those commonly implemented in deep learning frameworks (RNN, LSTM, GRU) are used with a more thorough examination of RNNs.

This investigation is of interest because often, reports of failures or issues are unpublished. Still, since the case described here is such a fundamental problem, it can be assumed that other researchers have or will face similar issues. At the very least, it might bring awareness to a problem type that is difficult to solve using the models that are the focus of this chapter.

In the following, first the related work towards modeling of dynamical systems will be shown, as well as past research on integrating sequences with recurrent networks. Afterwards, the problems that were experienced with recurrent networks will be described in depth. Mathematical proofs are conducted to show that there exist parameter constellations where all three recurrent layer types should be able to model the required system equation. From there on, the evolution of the model parameters during the training process is investigated. Based on this, the actual source of the problem is extracted.

4.1 Related Work

The following section aims to provide an overview of relevant methods of dynamical systems modeling, particularly in combination with machine learning. Then recurrent neural networks and their known issues are described.

Dynamical systems are concepts used to describe the behavior of a physical system based on its current state and external inputs. An example of modeling such a system using ordinary di erential equations (ODE) to predict the output of the system $\hat{y}(t)$ depending on external inputs $x(t)$ based on, e.g., a function f is

$$\frac{y(t)}{dt} = f(x(t)). \tag{4.2}$$

To compute $y(t)$, i.e., the output in the time domain, these equations are integrated using an ODE solver and starting value y_0 with the initial system state. Dynamical systems range from easy understand and analyze to more complex structures with chaotic properties. Simpler ones are e.g., linear time-invariant (LTI) systems. This means for Equation 4.2 that $y(t)$ is linear w. r. t. x(t), and the starting point of the simulation does not change the system's behavior (time-invariant). Non-linear systems are modeled e.g., by using lookup tables or non-linear functions as part of an ODE

[89]. The chosen equations depend on physical properties of the system's sub-components. The development of a model without prior data requires thorough background knowledge on numerical simulation itself as well as the system and its sub-components. If a systems' underlying parameters and structure are unknown, ML methods can be used to deduce them from measurement data of the system experimentally. Instead of known exact equations based on physical properties, generic equations and placeholder parameters are used, which are optimized based on real measurements of the system [90]. Because of this, models, for example for numerical simulation, often are available only late in the development process of hardware.

An example for ML based methods is Sparse Identification of Nonlinear Dynamics (SINDY) by Brunton et al. [89]. They demonstrate the learning of pre-defined non-linear di erential equations of a system using Least-Squares optimization. The equations are pre-defined based on known properties of dynamical systems, such as linear, quadratic, sine/cosine functions, and is therefore applicable when the behavioral equations of a system are known. The resulting equation are then integrated using an ODE solver. The authors demonstrate this on a Lorenz attractor and in fluid flow simulation. A problem for this approach is noisy data in the time domain, for which the authors advise the use of pre-filtering.

Another possibility of modeling systems based on ODE using ML are neural ordinary di erential equations (NODE) published by Chen et al. [91] in 2018. The system equations are modeled using deep neural networks. The models are used to predict the system's behavior in the time domain using an ODE solver. The advantage of this approach is, that the function can be sampled at di erent time intervals, raising the accuracy of the sampling if necessary. NODE are versatile models and a relatively new field of research. Wilczek et al. use Neural ODEs for the modeling of electrical circuits and achieve impressive accuracy regarding the original frequency response of the circuit [92]. Brucker et al. [93] demonstrate Neural ODEs for the modeling of lithium-ion batteries. Other pieces of work even use them for image segmentation [94] and 3D mesh generation from images [95]. Since RNNs will be shown to not be successful in modeling the task in this chapter, they are an interesting and promising subject for future research.

Since the starting value of the signals in the test bench are at a zero level in the hydraulic system under test, it is possible to model the system output $y(t)$ in a time-series to time-series (or sequence-to-sequence) model based on input channels $x(t)$ and training the model using gradient descent. This simplifies implementation and training. Therefore, recurrent neural networks were chosen originally. It was surprising to run into issues using these networks for simple tasks, such as integrating a longer sequence.

The first research on Recurrent Neural Networks (RNNs) was published 1985 by Rumelhart et al. [96] and 1986 by [97]. In 1997, [81] published their research on LSTM (Long-Short-Term Memory) in 1997. Another layer type that has shown similar in performance to LSTM is GRU (Gated Recurrent Unit) by Cho et al.[82] from 2014. In the following, Re-

current Network (RN) means the general class of these networks, while RNN, LSTM and GRU are implementation flavors.

RNs have proven successful in text- and sequence processing and prediction tasks. Other work has also shown their merits in regression problems concerning e. g. bacteria growth prediction [84], water level prediction of lakes [83] or finance prediction [98]. Yan et al proposed a LSTM autoencoder for anomaly detection in aircraft hydraulics [31]. For the domain of water hydrology and hydraulics, Zounemat-Kermani et al. reviewed their use and showed that recurrent networks model the desired behaviors accurately in di erent cases [99]. Regarding highly sampled time series, Chung et al. performed a study on speech signal synthesis and found LSTM and GRU performing similarly well and better than RNN [85]. Sak et al. also used LSTM networks for the modeling of acoustic sound data [100] and evaluate di erent multi-layer recurrent networks and claim state of the art performance. This means that for hydraulic data and for sequences with numerous samples, there exists related work that reports satisfactory results for all three recurrent layer types.

Related work exists that investigates the counting of elements or summation of a sequence for RNs. This problem type entails that model must be able to calculate a sum over given input data. At the same time, it must keep its current output unchanged when no relevant changes occur in the input (as an example see Equation 4.1). With regards to language processing, Rodriguez et al. [101] applied RNNs for stack free text analysis. They analyze the classification whether all open brackets in a text are also closed. They examined di erent recurrent network architectures under the standpoint of dynamical system analysis and encountered issues for modeling of sequences with a length over 32. This use case is di erent from the one in this chapter since it entails that the network counts and classifies the counted value. Suzgun et al. [102] evaluated di erent network architectures for counting objects in grammar and achieved satisfactory results using a single-layer LSTM for one of the tested languages (Dyck-1), while finding no suitable recurrent architecture for another (Dyck-2). Gers and Schmidhuber show that a modification to LSTM enables it to "time and count" in di erent data sets, claiming that standard LSTM can time for 50 steps in a sequence, and present a modification to LSTM to bridge even 1000 steps [103]. Elman et al. [104] also show a counting LSTM architecture, but the experiments used only have a small maximum count and do not explicitly mention very long-term memorization.

A known problem for RNs is vanishing or exploding gradients. Vanishing gradient means that the gradient of a parameter in a neural network becomes too small, and thereby it does not change with further iterations of the gradient descent. There are several reasons: the networks' non-linear activation function, often $\sigma(x)$ and $tanh$ can run into saturation with large and very low values of x. Bengio et al. [105] show that another reason is the length of the sequences, which cause values from the start of the sequence to have less influence on the final gradient of the model due to the backpropagation through time. They propose that one way to relieve this issue is using global optimizers instead of gradient descent

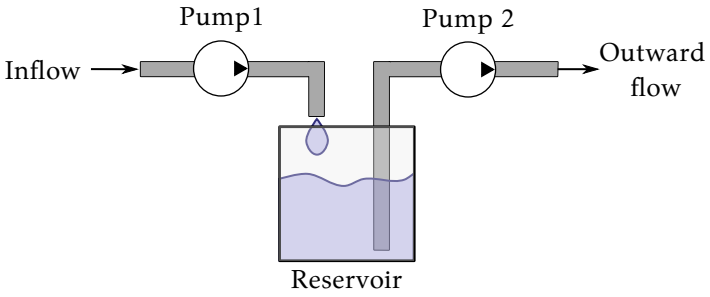


Figure 4.2: Hydraulic system that the given data set is based on.

or incorporating domain knowledge directly into the model, instead of relying on training. Exploding gradient mean suddenly large gradients during training. This can lead to large loss values, NaN loss values and to the end of the training process. Grosse [106] describes how this can occur for RNs due to the repeated application of the same function to an input. This is also called iterated function. More iterations lead to higher probability of chaotic behavior, and therefore failed training. There exist strategies against exploding gradients like gradient clipping, input reversal and identity. This is a good pointer towards our problem, because as it will be shown, for smaller sequences an RNN is able to learn the integration of a sequence, but with longer sequence length, the model can no longer reach the optimal parameters.

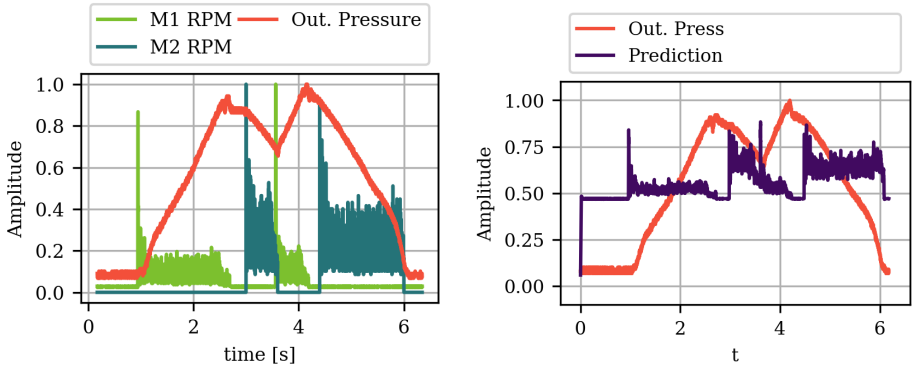
The following chapter is the first work to focus only on the integration problem for exceedingly long sequences.

4.2 Problem Description

This section will describe the challenges in modeling the integrating system. The data set this chapter is originally inspired by is recorded from a hydraulic system in a test bench described in Figure 4.2. A motor (M1) which actuates a hydraulic pump to fill a reservoir. A second motor (M2) then pumps out the volume. Only relevant signals are viewed in this context.

All channels in the data set are normalized to (0,1) prior to training. The sampling rate is 1kHz and one measurement duration is approximately 6 seconds. In Figure 4.3a, it can be seen that with the rising RPM (rotations per minute) of M1, the level in the chamber rises. With increased RPM of M2, the level decreases. Three di erent recordings were used for training, validation, and testing in this chapter.

One example of a failure case is a neural network with a RNN layer with 64 neurons combined with one fully connected layer. It was trained for 1000 epochs on the one training data set in a sequence-to-sequence approach for the complete time series. Figure 4.3b shows the output of the network on the training data. The output is not matched correctly. In the sections where the input is above minimum level, the network outputs an oscillating signal. An output like this could not be used for anomaly detection, because it does not even approximately match the systems actual



(a) Hydraulic data set. With M1 activation, the level in the reservoir increases. With the activation of M2, it decreases until empty.

(b) RNN Results with network with recurrent layer (64 units) with one fully connected layer

Figure 4.3: Example of the hydraulic data set used in this chapter and the prediction of a trained RNN-network.

output. The result shown is produced using an RNN layer, but LSTM and GRU run into similar problems, as shown later.

Based on the original data set an additional, simplified data set was designed to isolate the root cause of integrating/counting. It is described in Figure 4.4 and called "step" data set in the following. In this data set, there is only one input channel, the open or closed signal of a faucet. The output of the system is, like the original data, the fluid level or integral of the input, and is computed using Equation 4.1. Because of this, the data can be set to arbitrary length. The data can be seen in Figure 4.5. The data was sampled at 1ms.

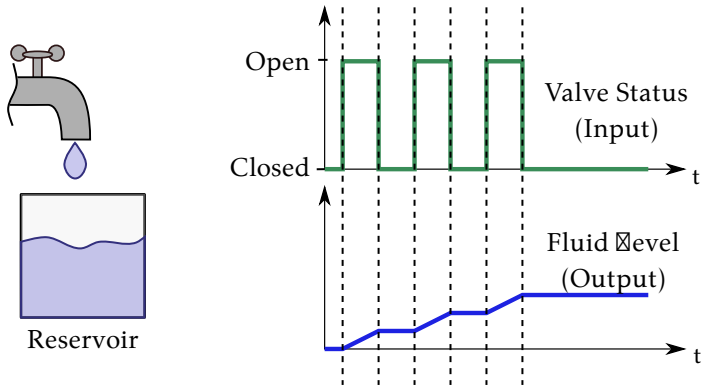


Figure 4.4: Simple example of system where summation over time is needed. With constant fluid flow, the fluid level in the container will increase with the amount of time where the valve is opened.

The results on the training data set after training for 1000 epochs each with each layer type (RNN, LSTM, GRU) are shown in Figure 4.6. The networks are not able to model the data correctly. Instead of an stepwise

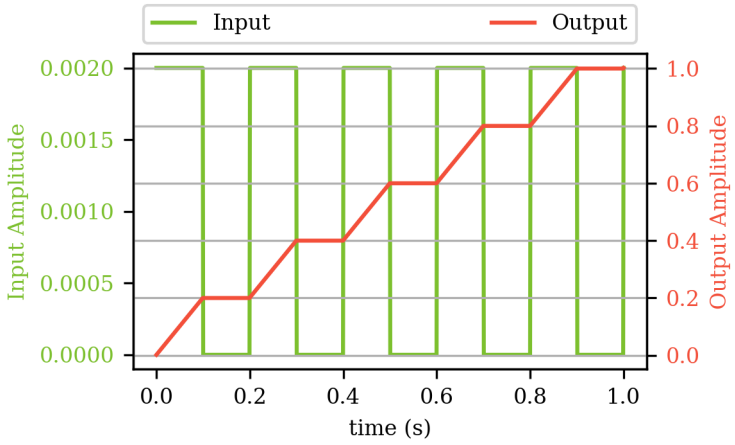
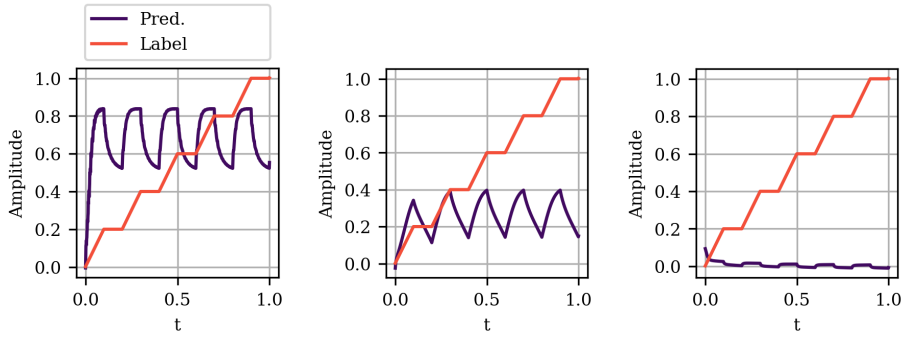


Figure 4.5: Minimal "step" example data set used in deep dive on the given problem, based on properties of exemplary system.

increasing function, the output is instead an oscillating function around a certain value, which has no similarity to the wanted output. This also shows how this data set is easier to visualize the core of the issue.



(a) RNN (b) LSTM (c) GRU

Figure 4.6: Results with network with di erent kinds of recurrent layers (64 units) and one fully connected layer.

4.3 Root Cause Analysis of Challenges

Several problem causes are investigated. First, it will be shown that, even with more neurons, layers, and the usage of CNN layers for pre-filtering, the recurrent neuron types do not learn the hydraulic data set described before. This leads to a closer look at the equations of the recurrent networks. It will be shown that in theory, there exist parameter solutions for each di erent layer type to achieve the goal model. Finally, the traversal of the parameters of an RNN neuron through the solution space is shown for one short and one sequence of original length, to demonstrate that the sequence length poses the main issue with this problem.

Implementation was done with Tensorflow [107] version 2.3. If not mentioned otherwise, MSE is used as loss function and as optimizer, the Adam-optimizer [108] with a learning rate of 0.001 was used in all experiments.

4.3.1 Layer and Neuron Count

To find if this problem applies for Recurrent networks in general for this kind of data, first, several di erent network configurations are assessed. First, the influence of the amounts of neurons and layers is examined. This is to see, whether the performance of the recurrent layers can be improved by adding more parameters to the model. The original data set of a hydraulic system is used. Three di erent amounts of layers and neurons are evaluated for each of the recurrent layer types, without CNNs. The models were trained for 1000 epochs on the training data set using the Adam optimizer with a learning rate of 0.2. The results for all combinations can be seen as R^2 scores in the Tables 4.2a through 4.2c. In the tables, scores above 0.5 are highlighted. For prediction examples for the di erent scores see Figure 4.7. RNN and LSTM models did not achieve a R^2 score above 0.5. With GRU, some models were able to pass the threshold, but there is no clear trend on whether more layers or more neurons improve the results. Therefore, it is difficult to pose a general rule on how to structure a network, that reliably models everything correctly. Due to the noisy nature of the data, next the influence of CNN layers for pre-filtering are discussed.

Table 4.1: Results for di erent network combinations. Bold cells are where the output was of similar shape as the input (see Figure 4.7)

(a) RNN

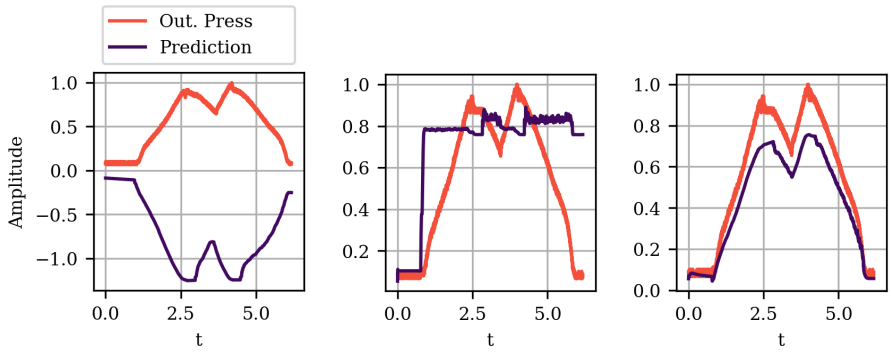
	Neurons		
Layers	1	16	32
1	-12.91	-5.85	-0.08
2	-0.74	-10.91	-2.88
3	-8.00	-0.15	-12.54

(b) LSTM

	Neurons		
Layers	1	16	32
1	-1.21	-8.11	-0.88
2	-17.83	-0.68	0.14
3	-20.72	-8.83	-0.62

(c) GRU

	Neurons		
Layers	1	16	32
1	-22.42	-0.80	**0.64**
2	-5.85	-2.82	**0.51**
3	-1.14	**0.84**	-22.17



(a) GRU with R^2 = -22.41 (b) LSTM R^2 = 0.14 (c) GRU with R^2 = 0.84

Figure 4.7: Example results and R^2 scores. R^2 score above 0.5 was counted as successful.

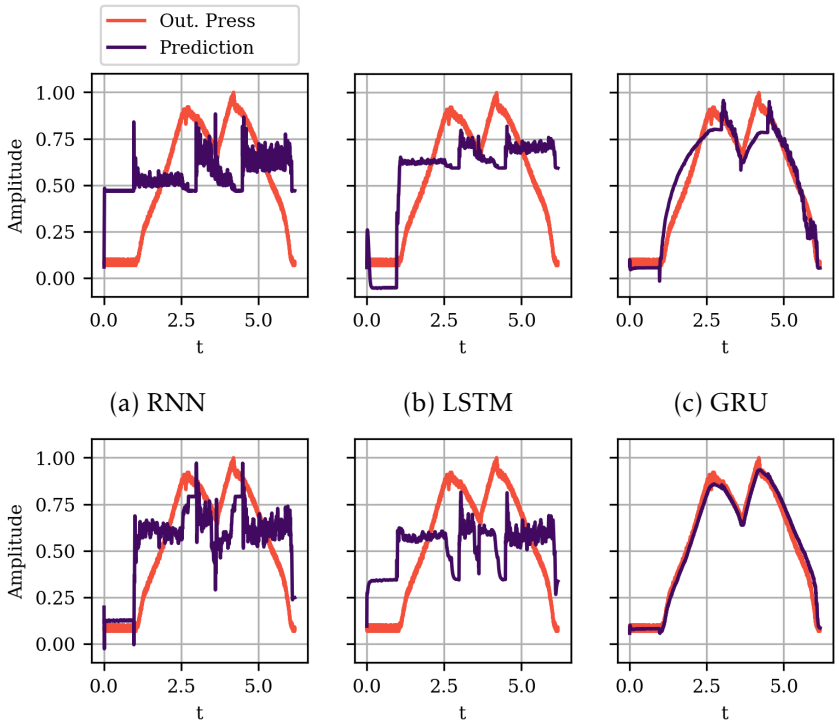
4.3.2 Neural Network Architecture Variants

Since there was no obvious pattern on how to structure RN-only networks to reproduce the hydraulic data set, the influence of CNNs is examined. The motivation in using CNNs is the fact, that they are structured like discrete signal filters. Therefore, they could function as an additional support for the following recurrent layers by e.g., removing noise. The three layer types are evaluated without and with a first CNN layer for filtering of the data on the hydraulic data set. First, a recurrent layer (RNN, LSTM, or GRU) is trained on its own with a neuron count of 125 followed by a fully connected layer with one neuron. The second network architecture assessed is a convolutional layer with a filter count of 25 and a kernel size of 25 followed by the recurrent layer and a one-neuron fully connected neuron. Training is done for 1000 epochs each with the Adam optimizer with a learning rate of 0.2. Training data was one example of the measurements from the hydraulic system as shown in Figure 4.3.

The results on the testing data are shown in Figure 4.8. There is no visible di erence for RNN and LSTM with or without the CNN. The result of the CNN with GRU model is noticeably better than the others (Figure 4.8f).

This contrasts with Section 4.2, where it was shown that for noise free rectangular data set, a GRU network also does not converge on a satisfactory solution. Therefore, there is the question whether the superior results of the CNN+GRU model on the hydraulic data were achieved due to overfitting, where the general movement of the output is memorized by the model without "understanding" the systems properties correctly. For this, a modified hydraulic data set is used, where half of the recorded time, the input data is set to zero. Figure 4.9 shows the results for this.

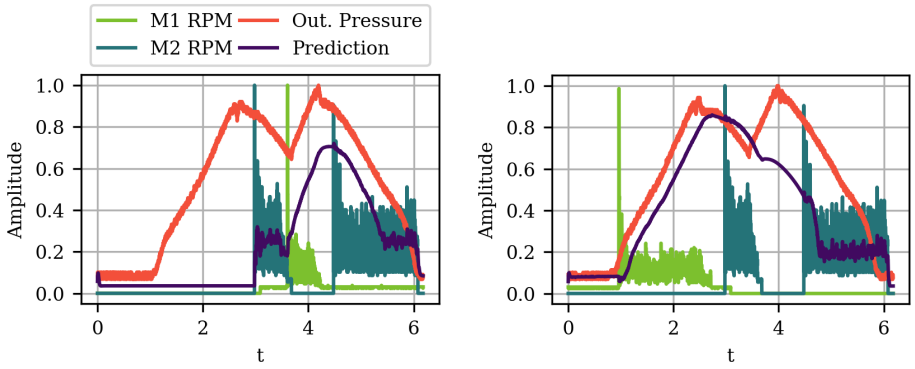
In Figure 4.9a, it can be seen that the model correctly does not generate the first "hump" in the output. Still, going from the assumption that the orange signal M2 acts as a negative factor in the output, a downward slope would have been expected in the time range of $3 < t < 3.7$. Otherwise, the model reacts properly to the M1 input by increasing the output signal



(a) RNN (b) LSTM (c) GRU

(d) CNN + RNN (e) CNN + LSTM (f) CNN + GRU

Figure 4.8: Results of recurrent networks and with and without one convolutional layer as the input layer.



(a) First half of input omitted. (b) Second half of input omitted.

Figure 4.9: Results with network with one convolutional, one GRU layer (64 units) and one fully connected layer for modified input data set. The model can approximate the behavior of the system.

in $3.7 < t < 4.4$. The output decreases again with the second half of the orange input at $4.4 < t < 5.5$, but not all the way back to zero level, which is incorrect.

Figure 4.9b shows the results for the data set, where the second half of the "M1 RPM" input is omitted. The output increases and decreases correctly in the first half ($0 < t < 3.8$) but decreases further even though no "M2 RPM" signal is present in $3.8 < t < 4.5$. For the second half of the "M2 RPM" input, at $4.5 < t < 6$, the output signal remains constant, which is also not according to the actual system's properties.

Therefore, even though the initial results of the CNN+GRU combination seemed to show an improvement, the model did not learn the complete system's behavior correctly. More diverse training data sets might improve the results. But since training data often is limited, it is important that the dynamics of a system can be learnt based on few examples.

Therefore, the next section more thoroughly examines the neuron's parameters, and looks at the weight combinations that would be necessary for the task.

4.3.3 Possible Neuron Parameter Configurations

Based on di erent architectures with unsatisfying results, the next question to solve is, whether this can even be learnt by the recurrent neurons used, which internally rely upon non-linear functions.

In the discrete case, the integral of a time series over time, i.e., its cumulative sum as a sequence-to-sequence translation problem, is computed using the recursive Equation 4.3:

$$y(t) = y(t-1) + K \cdot x(t), \tag{4.3}$$

With $y(t)$ as the output at a certain time step, $y(t-1)$ the output at the prior time step and $x(t)$ the current input value, K is a constant factor. In the following, it will be shown that there exist solutions for the Equation 4.1 for all recurrent layer types discussed. The proofs will be shown under the assumption that the input and output values are in a small range and on architectures with one layer with one neuron for each of the layer types. This simplification can be valid for other data ranges if a fully connected layer is appended to the recurrent layer, since it could function as a scaling factor for the networks output.

The activation functions used internally in all three layer types are $\tanh(x)$ and $\sigma(x)$. For these functions, the following assumptions for small input values x with $| x | < 0.5$ the following approximation can be made (Figure 4.10):

$$tanh(x) \approx x \tag{4.4}$$

$$\sigma(x) \approx 0.25x + 0.5 \tag{4.5}$$

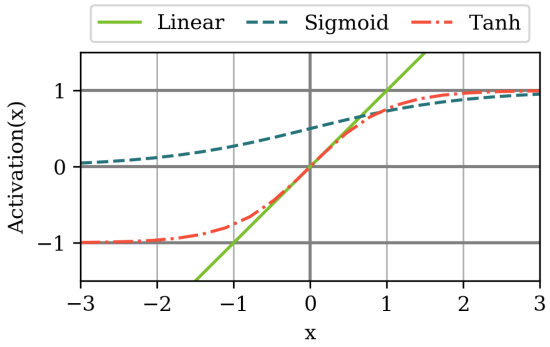


Figure 4.10: Common activation functions in RNN: Linear, Sigmoid and Tanh.

RNN

The implementation of the SimpleRNN Layer in Tensorflow follows the following computation:

$$h_t = a(W_x x_t + W_h h_{t-1} + b), \tag{4.6}$$

with $a(x)$ as the activation function (common default: $\tanh(x)$ or $\sigma(x)$), W as weight matrices and b as a bias vector. h is the hidden layer result and output of the unit (see also Figure 4.11). Under the assumption that a linear activation function is chosen, the configuration to match Equation 4.1 with $K = 1$ is achieved when:

$$W_x = W_h = 1 \tag{4.7}$$

$$b = 0. \tag{4.8}$$

Similarly, if the tanh activation function is used, this is also true for small values x in $tanh(x)$, since $tanh$ behaves approximately linear in the range $(-0.5, 0.5)$.

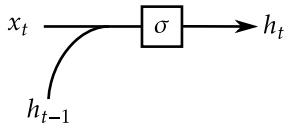


Figure 4.11: Standard RNN architecture as defined in Tensorflow [107]. The current input value is combined with the output of the layer on the last output before being passed to the activation function.

LSTM

LSTM units consist of three gates (input gate i_t, forget gate f_t, output gate o_t) which combine the result on the prior time step, i.e., h_{t-1} with the current time step's input x_t to compute the output h_t. Additionally, LSTMs

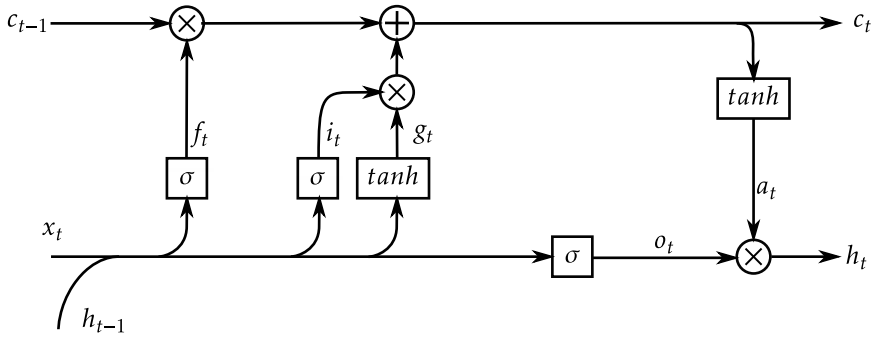


Figure 4.12: Standard LSTM architecture, the current input value is combined with the output and the memory on the prior data point before being passed to activation functions.

carry a term c_t (memory cell) meant for the encoding of long-term memory across time steps. This is shown in Figure 4.12.

The following equations are computed inside an LSTM layer [81]:

$$f_t = \sigma(W_{fx}x_t + W_{fh}h_{t-1} + b_f) \tag{4.9}$$
$$i_t = \sigma(W_{ix}x_t + W_{ih}h_{t-1} + b_i) \tag{4.10}$$
$$g_t = \tanh(W_{gx}x_t + W_{gh}h_{t-1} + b_g) \tag{4.11}$$
$$c_t = f_t \circ c_{t-1} + i_t \circ g_t \tag{4.12}$$
$$o_t = \sigma(W_{ox}x_t + W_{oh}h_{t-1} + b_o) \tag{4.13}$$
$$a_t = \tanh(c_t) \tag{4.14}$$
$$h_t = o_t \circ a_t \tag{4.15}$$

Here, $\circ$ denotes the Hadamard Product operator. One solution towards the counting of this unit is this parameter configuration:

$$b_f \longrightarrow -\infty; \qquad |\, W_{xf,hf} \,| << b_f \tag{4.16}$$
$$b_i \longrightarrow \infty; \qquad |\, W_{xi,hi} \,| << b_i \tag{4.17}$$
$$b_g \longrightarrow 0; \qquad W_{xg,hg} = 1 \tag{4.18}$$
$$b_c \longrightarrow 0; \qquad W_{xc,hc} = 1 \tag{4.19}$$
$$b_o \longrightarrow \infty; \qquad |\, W_{xo,ho} \,| << b_o \tag{4.20}$$

This solution, due to the approximations in Equation 4.4 and 4.5, leads to

$$f_t \to 0 \tag{4.21}$$
$$i_t \to h_{t-1} \tag{4.22}$$
$$g_t \to 1, \tag{4.23}$$

and therefore

$$a_t \to x_t + h_{t-1} \tag{4.24}$$
$$o_t \to 1. \tag{4.25}$$

This leads to the final output of the unit of

$$h_t = x_t + h_{t-1}, \tag{4.26}$$

by which Equation 4.1 for $K = 1$ is achieved. This configuration under the described weights results in the network shown in Figure 4.13. Note that the gates often must reach either very small or very large values for the unit to reach the needed functionality. Still, there exists a solution to the given problem with one LSTM neuron in one layer.

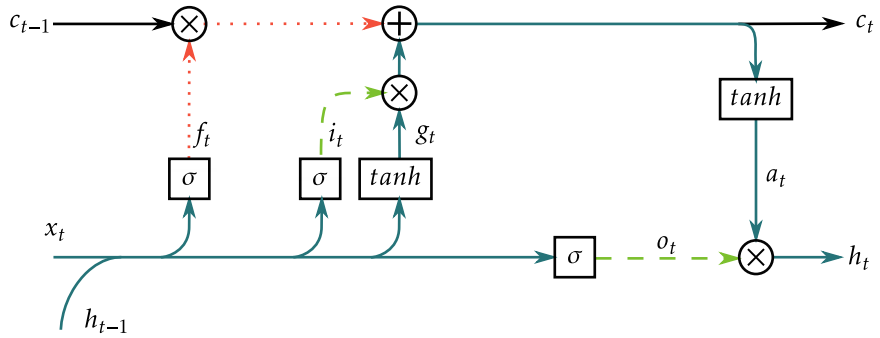


Figure 4.13: One LSTM configuration for integration. The dashed lines (green) mean that the value passed should approach 1, while the dotted (red) are pushed towards 0.

GRU

A GRU unit's architecture is defined as shown in Figure 4.14 and described by [82] as

$$z_t = \sigma(W_{zx}x_t + W_{zh}h_{t-1} + b_z) \tag{4.27}$$
$$r_t = \sigma(W_{rx}x_t + W_{rh}h_{t-1} + b_r) \tag{4.28}$$
$$\widetilde{h_t} = \tanh(W_{hx}x_t + W_{hh}(r_t \circ h_{t-1}) + b_h) \tag{4.29}$$
$$h_t = LI(h_{t-1}, \widetilde{h_t}, z_t) = (1 - z_t) \circ h_{t-1} + z_t \circ \widetilde{h_t}. \tag{4.30}$$

LI stands for linear interpolation between h_{t-1} and $\widetilde{h_t}$ parameterized by z_t.

Given the approximations Equation 4.4 and 4.5, one solution to achieving the integration in Equation 4.1 is a configuration where

$$\widetilde{h_t} = h_{t-1} + x_t \tag{4.31}$$
$$z_t = 1, \tag{4.32}$$

which means $\widetilde{h_t}$ contains the desired result, and at the same time the upper lane with h_{t-1} is multiplied by zero. $r_t \to 1$ has to hold, to pass h_{t-1} through to $\widetilde{h_t}$ unchanged. This means that the parameters are

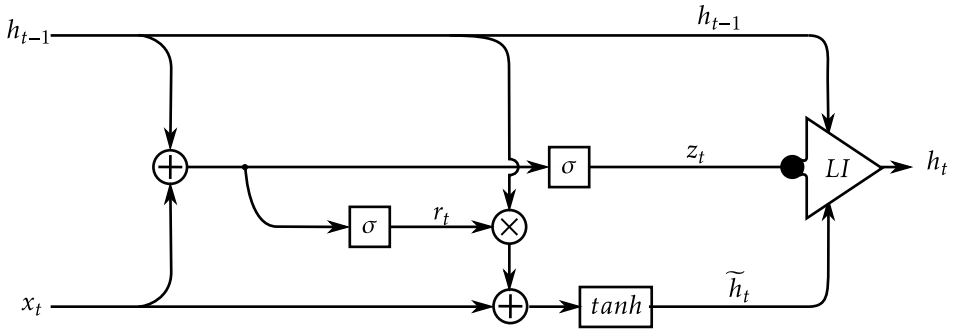


Figure 4.14: Standard GRU architecture as defined by [82]. The LI-node denotes the linear interpolation of h_{t-1} and $\widetilde{h}_t$ parameterized by z_t.

$$W_{zx} = 0, W_{zh} = 0, b_z \to -\infty \tag{4.33}$$
$$W_{rx} = 0, W_{rh} = 0, b_r \to \infty \tag{4.34}$$
$$W_{hx} = 1, W_{hh} = 1, b_h \to 0. \tag{4.35}$$
$$\tag{4.36}$$

The resulting setup is shown in Figure 4.15. This means, again, that the unit must assume a very specific parameter configuration to reach the wanted output.

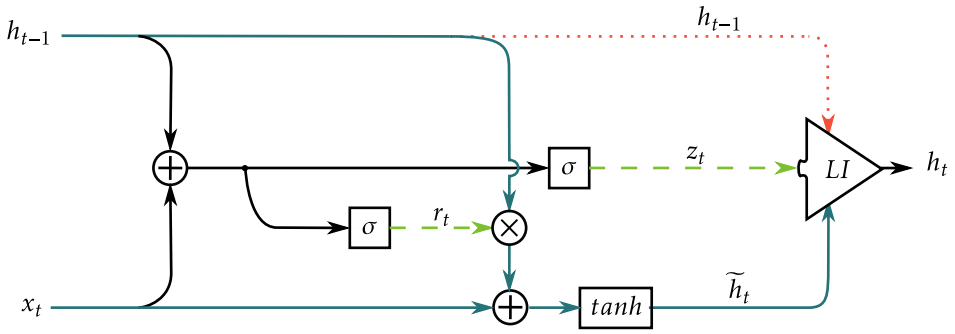


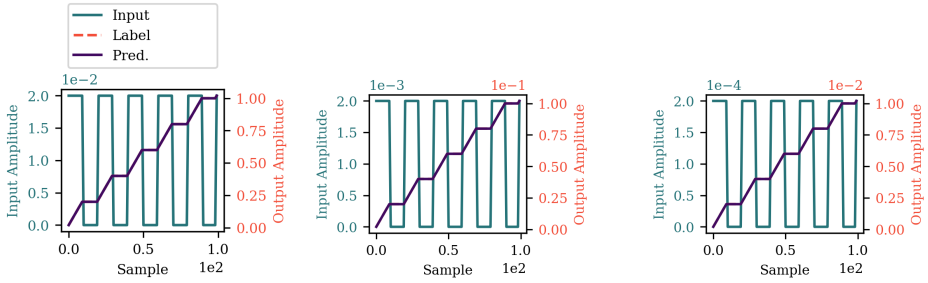
Figure 4.15: One GRU configuration for integration. The dashed lines (green) mean that the value passed should approach 1, while the dotted (red) are pushed towards 0.

Having shown that there exist solutions for all three layer types, these configurations will be verified for di erent scaling of data by using them on the "step" data set.

Experiment

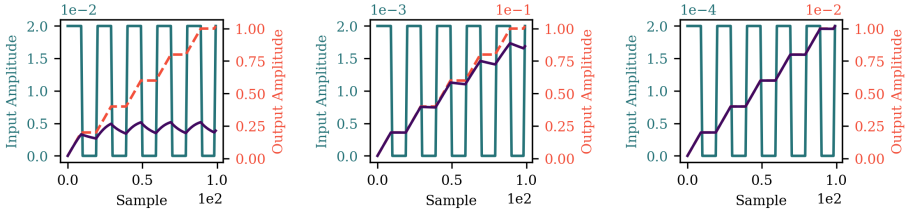
It was shown that for RNN, LSTM and GRU, there exists a solution for the integration problem. It was also shown, that the found solutions only are valid in a certain value range. In this section, this will be verified.

For this, the artificial step data set with length 100 is used at di erent data scales (0.01, 0.001 and 0.0001). Each network consists of one layer with one neuron. For the RNN, the linear activation was chosen, where it was known that it would be able to approximate the wanted behavior. For each parameter of the networks, the optimum values described in the proofs are pre-assigned, instead of using training with gradient descent optimization. The results are shown in Figure 4.16 through 4.18.



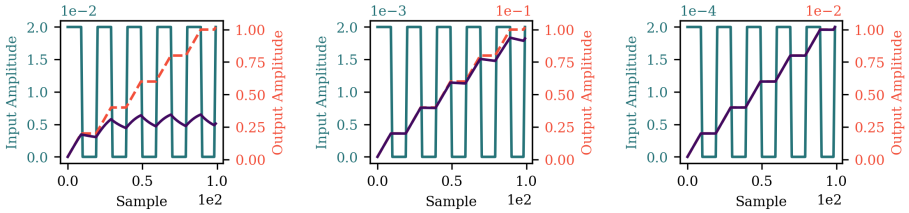
(a) scale=0.01 (b) scale=0.001 (c) scale=0.0001

Figure 4.16: Results of RNN with linear activation function. Regardless of the scaling, the data is modeled accurately



(a) scale=0.01 (b) scale=0.001 (c) scale=0.0001

Figure 4.17: Results of LSTM with the proposed parameters shows dependence on output data scale. Only with a small scale, the data can be modeled accurately



(a) scale=0.01 (b) scale=0.001 (c) scale=0.0001

Figure 4.18: Results of GRU with the proposed parameters shows dependence on output scale. Only with a small scale, the data can be modeled accurately

As expected, the RNN with a linear activation can model the output correctly, regardless of data scale (Figure 4.16). For LSTM and GRU, the results are in Figure 4.17 and 4.18, respectively. Here, the smaller the signal range, the better the model approximates the wanted signal. The results of both are similar.

Normally, data for machine learning applications is normalized to a range between 0 and 1. The optimum value range for this problem is therefore comparatively small. If a linear layer were to be appended to the recurrent layer, this should in theory function as a trainable scaling factor and enable the model to fit the data successfully. Still, as shown in the prior experiments, in practice this does not happen, leading to incorrect models.

Therefore, next the traversal of weights in the RNN neuron during training are investigated.

4.3.4 Traversal of Weights Through Training

As a last step towards finding the root cause of the problem, the actual traversal of parameters of a RNN layer is examined. It was shown prior that the linear RNN unit can model the integration regardless of data scale in Section 4.3.3. Since vanishing gradients are either attributed to sequence length or the activation function in related work, a RNN unit is trained with a linear activation function. This section focuses on RNN, because it has only three parameters and can therefore be visualized easily in a 3D plot. Furthermore, the simple structure of the neuron makes the actual issue more understandable.

As data sets, the "step" data set with 100 timesteps and with 1000 timesteps is used. Five training runs are conducted each. For the short data set, the training is run for 1000 epochs, for the long data set for 5000 epochs. In both cases the weights are initialized randomly. The Adam optimizer with a steeper learning rate of 0.01 was chosen.

The traversal of the weights is shown in a 3D volume of the models' loss. The volume is colorized using the viridis colormap, which is a colormap from yellow to green from small to big values. Each axis of the volume is one of the RNN's parameters w_i, w_h, b. The parameters of the RNN units at the start of the training are shown as a cross. The optimum weight constellations of the neuron based on Section 4.3.3 are known and painted into the volume as a dark blue circle.

Results for the short data set are shown in Figure 4.19 and in Figure 4.20 for the long data set. For the short data set, four out of five neurons meet the optimum after performing a detour towards the positive b-direction. For the long data set, all points also traverse into the positive b direction first, but they then perform a curve towards the large negative w_x direction until approximately a w_x of -20. Then they approach the direction of the optimum, but randomly circle backwards into a backwards loop. One of the neurons jumps towards the negative b-direction before finishing training. The jumps cause an increase in the loss of the model. In the end, none of the models meet the optimum for the long data set. It seems like there is an insurmountable "wall" surrounding the optimum, that the parameters cannot reach.

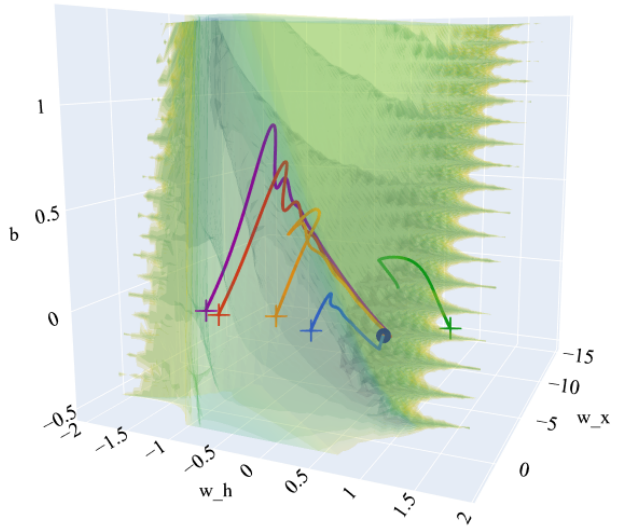


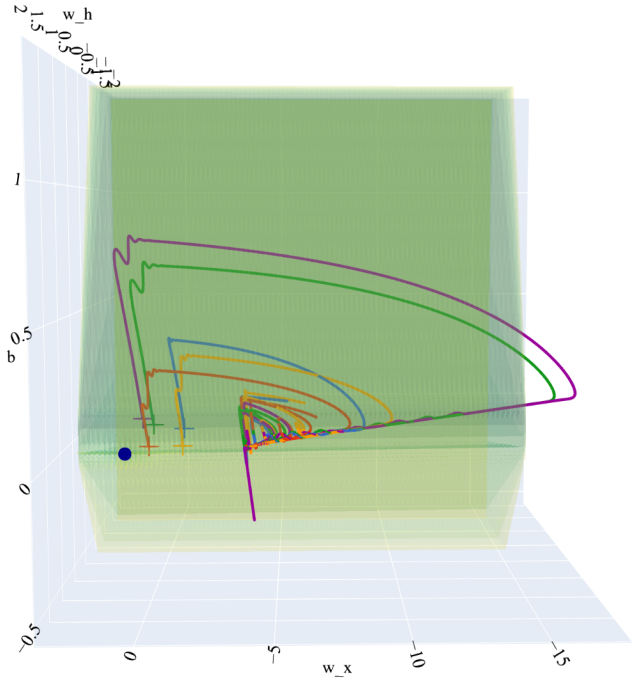
Figure 4.19: Traversal of RNN unit weights for the shorter input sequence. The crosses indicate the start of the traversal. The optimum is achieved with all 4/5 points in this experiment.

4.4 Interpretation of Results

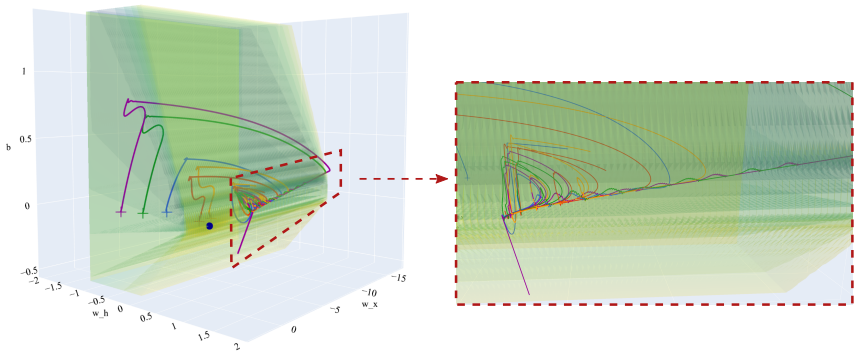
Interpreting the results, two possible explanations for this movement of the weights can be o ered. One cause that can be ruled out are the non-linear activation functions since the activation function was chosen to be linear. First, there is the instability of the model. The BIBO (bounded input, bounded output) principle of stability in signal processing postulates that a system is stable if there is an upper bound of its outputs for every point in time [109]. For the function that must be learnt, there is no upper bound if the input is not limited. This means that the model also becomes more unstable, the closer it gets to the desired functionality. Recurrent neural networks reapply the same function iteratively on the input sequence. This means that for long sequences, small errors in the weights of the model influence the output increasingly. For exceedingly long sequences, especially for the counting problem, small errors sum up to very large errors over the complete sequence which are, due to the definition of the problem, not bounded.

Second is the training process itself. It seems like, due to the step size of the gradient descent, there are "miss-steps" into a wrong trajectory (the loop-back), similar to taking the wrong exit on a highway. This means that the parameters first go backwards in the shape a loop. Since all trajectories landed inside this loop, the chances for taking the correct "exit" are slim. This miss-step into the loop might occur once the optimizer automatically increases the step size if the gradient changes too slowly. The reason for the large loop back might be due to the instability of the task. This can be the reason for the gradient "wall" that is shown for the long sequence.

To conclude, even without a non-linear activation function, recurrent networks cannot learn integration of a very long sequence of the shown



(a) View on $b - w_x$-plane



(b) Side view and zoom in on loopings.

Figure 4.20: Traversal of RNN unit weights with the long input sequence. The crosses indicate the start of the traversal. None of the starting points achieve the optimum.

shape. The long sequence length causes the errors of the model over time to increase, which brings instability during training. Since the integration problem inherently leads to theoretically unbounded outputs, it results in unstable models. This means that on the one hand, the problem is the length of the sequence. On the other hand, the problem is the task, i.e., the integration of the input sequence over time.

Bengio et al. [105] propose multiple solutions for this. One is simulated annealing. This means that several random starting points for gradient descent are chosen, and the descent is terminated if the gradient converges. Afterwards, the best result is chosen. This does not feasible for this data set. As can be seen, only one parameter configuration for the RNN is possible the learn the wanted behavior, and aside from the optimum point the greater part of the remaining points causes infinite loss of the model. The solution space is sparse. Due to the long sequence length, even small deviations in the weights from the optimum result in high prediction losses. Their other proposal, to use pre-parameterized equations, if possible, seems more promising. This falls under the domain of hybrid models, where simulation models are combined with ML methods.

In total, the seemingly simple problem is more complex than initially expected. Since the goal of this thesis is the anomaly detection for hydraulics – and not the modeling of hydraulic systems – this is the point where the root cause analysis and the pursuit of a model-based approach is concluded.

4.5 Chapter Summary

This chapter investigated the use of recurrent neural networks for the modeling of a common behavior of a hydraulic system, namely the integration of an input over time. Had this been achievable easily, there could have been a way to design digital twins for anomaly detection, the latter of which would have been the next focus of this thesis. Instead, opposite to expectation, the wanted behavior of the models was not achievable. Therefore, the underlying reasons were investigated instead.

For this, an easily interpretable data set was designed and used for evaluating several di erent network architectures. The problem of vanishing gradients and stability of recurrent models was investigated further with a focus on the length of the data set itself. Experiments were conducted to visualize the traversal of weights through the parameter space. This showed that for short sequences, the optimum was reached in most cases observed. For long sequences, the problem was two-fold: First, the gradient of the loss decreases over time due to vanishing gradients. Second: the summation of the recurrent model's error over the length of the sequence causes even smaller errors to influence the output of the network more significantly just due to the sequence length, which is why small noise leads to significant instabilities during training.

Machine learning and especially deep learning are popular research topics now. Still, there are problems that cannot be solved easily just with better hardware and larger models. This chapter showed such an example

of a seemingly trivial problem. Future work and more research should be conducted on this topic, for example, at which maximum sequence length the training of such a model is still reliably possible, and whether other layer types would be more robust for this type of problem.

Based on the findings in this chapter, the model-based approach is not pursued further in this thesis. Instead, the following research will be directed towards data-driven methods for anomaly detection.

5 Data-based Unsupervised Anomaly Detection for Test Bench Data*

In the last chapter, it was shown that a Digital Twin-based anomaly detection approach poses disadvantages due to the modeling e ort involved. Therefore, this chapter will focus on unsupervised, data-based anomaly detection methods instead. This is relevant because often, there is no labeled data before the start of the test bench. The evaluated methods leverage the domain knowledge on the use case, which yields a more specialized, but also more efficient solution than the model-based approach shown prior.

First, a description on the use case and problem statement. Test benches run continuously over long amounts of time and record with high sampling rates (in the data set used up to 100 Hz, but up to 10 kHz can be common). The data flow from the test bench to the experts' user interface is shown in Figure 5.1. The data usually is collected centrally and then processed further, e.g., for anomaly detection. The results are visualized in a dashboard for easy access. Viewing all the data directly (e.g., as shown in Chapter 3) is difficult because of the large amounts of measurements. Therefore, it is important to automatically detect when one of the systems under test shows extraordinary changes compared to the others. This is made difficult by the fact that tested systems often are first-of-their-kind prototypes, which means that no labeled training data for the development of anomaly detection systems exist. This necessitates an unsupervised approach [44].

Several devices are tested at the same time to produce a statistically significant reliability evaluation. The input values (pressure, voltage, temperature) to the test benches are pre-programmed, which is why the test bench output data is similar between measurement types and samples, if all of test subjects behave normally and are recorded at the similar points in the test lifecycle. The input and output data can therefore be seen as periodic time series. The investigated methods are motivated by the visualizations in Chapter 3 of repeating data. To recall, the visualization was efficient, because it aligned the measurements by time and amplitude, making abnormal data points more prominently visible.

This principle will be leveraged by the algorithm proposed in this chapter. For this, it is assumed that the systems under test function normally, on average. The "normal" maneuver of a certain time interval is

*Parts of this chapter are published in "D. Neufeld and U. Schmid, "Anomaly Detection for Hydraulic Systems under Test", in *IEEE 26th International Conference on Emerging Technologies and Factory Automation (ETFA'21)*, (Vasteras, Sweden), IEEE, 2021, pp. 1–8, ISBN: 978-1-7281-2989-1. DOI: 10.1109/ETFA45728.2021.9613265".

computed from the median value of all repetitions over time. The distance of every maneuver towards the "normal" is then classified using an unsupervised anomaly detection algorithm.

Six time series distance metrics and two different anomaly classification methods are evaluated. Furthermore, the robustness of the proposed method concerning the concept drift due to wear in the system is investigated.

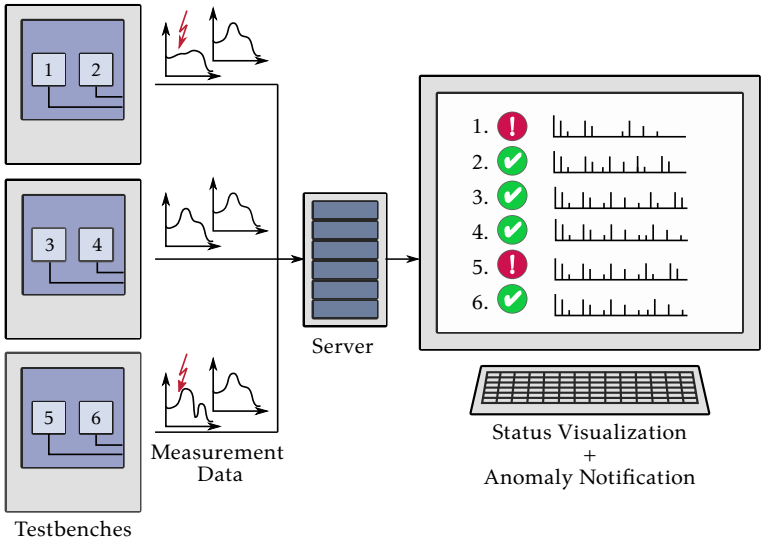


Figure 5.1: Schematic of data flow from multiple test benches with six total tested systems. The data is collected centrally and processed for easy access to technicians and engineers.

The following content first presents related work. Afterwards, the data set and methods used are described. The approach is evaluated using experiments.

5.1 Related Work

This chapter focuses on multivariate, periodic measurement signals under concept drift. The load applied to the system is pre-programmed by an engineer in test maneuvers, which means that the time series are time aligned and there are no periodicity changes between repetitions. Therefore, difference metrics for periodic time series and outlier detection algorithms are evaluated. The data set stems from work by Helwig et al. [8] and has been analyzed and described by them further in [59] and [32]. Their work focuses on accurate supervised classification of hydraulic system defects based on extensive feature engineering and feature reduction for effective search in time series data. This use case differs from the one in this chapter, which is to detect anomalies with as little prior feature engineering as possible, in an unsupervised way. Premature feature selection can cause errors in a system to go undetected, if they appear in feature spaces that were omitted before.

Anomaly detection for time series is a common topic in research (see e.g., the survey by Chandola et al. [25]). In terms of unsupervised anomaly

detection, for example in periodic ECG data, Chakraborty et al. [58] first reconstructed a ”normal” signal as an average of several repetitions using Dynamic Time Warp. The distance between a new repetition and the "normal" is computed and if it exceeds a pre-defined threshold, it is classified as an outlier. Manually determining a threshold is not desirable for this use case, because it reduces the flexibility of the approach towards new device types. In contrast, Hochenbaum et al. at Twitter published an approach for univariate detection on periodic data with an automatic threshold, which is specialized on point anomalies [34]. The time series is first decomposed with Seasonal Trend Decomposition (STD) into a trend, a seasonal and a residual component and the residual is then classified for anomalies using the Generalized Extreme Studentized Deviate Test (ESD Test) by Rösner [110], which uses the Student's t-distribution. The use case in this chapter deals with complete, multivariate time series instead of point anomalies. In experiments, Rösner shows that the ESD test shows reasonable accuracy at a sample size above $n = 15$. For point anomalies, this is not an issue, but it can be a limit for anomaly detection on complete time series for test benches with lesser samples, like in the dataset used in this chapter. Therefore, a di erent statistical measure, the Modified z-Score by Iglewicz and Hoaglin [111] is used here. Additionally, Local Outlier Factor (LOF), a nearest neighbor-based method, is evaluated. It has the advantage over other methods like the One Class Support Vector machine, that it can be applied without prior labeled training data [112].

Anomaly detection algorithms rely upon distance metrics of data points for classification. For time series distance metrics, prior reviews exist, for example by Serrà and Arcos [113] or Toller et al. [114]. Because of the high sampling rate and size of the data set, computationally efficient metrics are focused on in this chapter, which is why Dynamic Time Warp distance, for example, will not be part of this evaluation but rather, metrics like the Mean Absolute Error and Mean Squared Error.

With the rising popularity of deep learning, other anomaly detection methods, for example using Auto-Encoder networks have been applied either on time series or on time series features. Chalapathy et al. list several in their survey, among which use cases like industrial and IoT (internet of things) anomaly detection methods [115]. Additionally, Braei and Wagner [116] surveyed neural network architectures for anomaly detection in univariate series.

Audibert et al. [117] show deep learning methods for the unsupervised multivariate time series anomaly detection for IT systems, and Yin et al. [118] for IOT time series. Ding et al., for example, demonstrate the use of Auto-Encoders for anomaly detection in cyber physical systems using LSTM (Long Short-Term Memory) Auto-Encoders [119]. Auto-encoders function as unsupervised, non-linear feature extractors. Once trained, these neural networks can be used efficiently on modern hardware. For anomaly detection, the reconstruction error of the network is used as the distance metric. In general, these methods need a pre-set threshold for outlier classification. Alternatively, Hundman et al. demonstrate the use of a dynamic threshold in combination with LSTM auto-encoders [120].

If there are continuous changes in the system that cause a concept drift in the data, an Auto-Encoder model needs to be re-trained, or the anomaly threshold must be adjusted. For non-periodic data sets in more complex use cases, Auto-Encoder networks can be beneficial. Since test bench data often is periodic, this chapter aims to provide a more direct approach for anomaly quantification, as described in the next section.

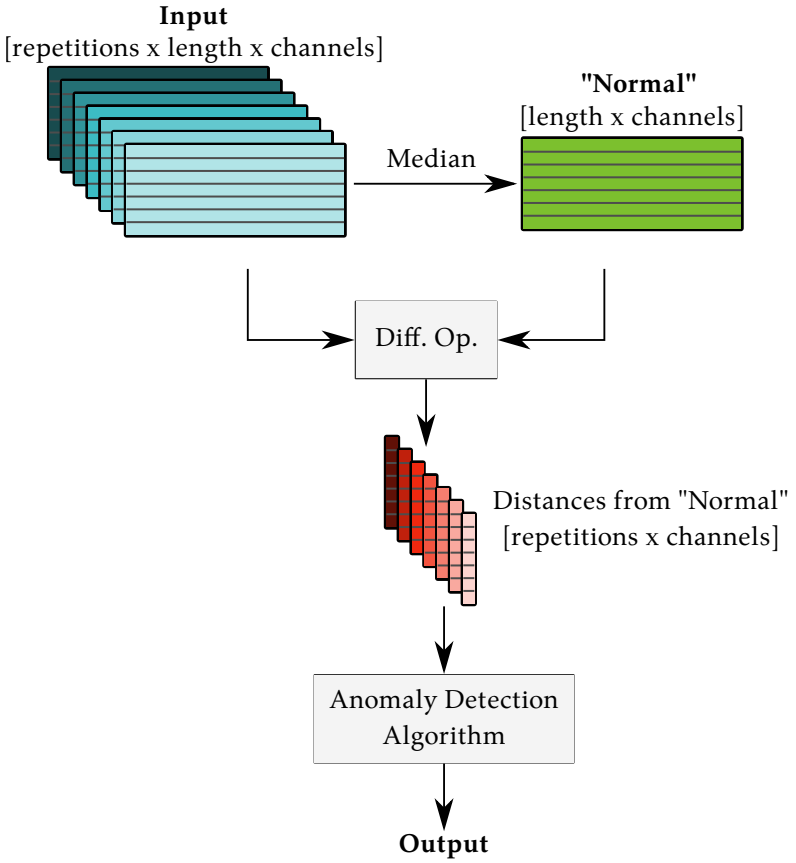


Figure 5.2: Flowchart of the proposed anomaly detection algorithm.

5.2 Median-Based Anomaly Detection

First, the method and the used di erence metrics are described, then the classifier methods. Afterwards, the method's interpretability and its application to data under concept drift are looked at.

5.2.1 Di erence Metrics

The maneuver repetitions in the data set are synchronized over time. Should this not be the case, for example because of measurement hardware limitations, the methods presented in this chapter can be applied after re-fitting the data on the time axis (as shown in Chapter 3), for example using Dynamic Time Warp or cross-correlation. Additionally, and it is assumed that all samples in the test bench are at the same point in time of the test run, which means that if all samples behave normally, their output

should be classified as normal. To deal with the unsupervised setting of the problem, the median value of all repetitions is assumed as the normal; and repetitions with unusually high deviation from it are counted as abnormal. Therefore, the anomaly detection process is divided into the following steps: First, the median maneuver is computed out of all repetitions. Then, the deviation from the normal is computed per channel for all repetitions, which are then used as input for the anomaly classification algorithm. This means that for this data set with 17 channels, each repetition yields a feature vector with 17 elements comprised of the di erences towards the median, which is used as input for the classifier. The median was chosen as averaging metric due to its robustness towards outliers. In the following, the evaluated metrics are described:

- Mean Absolute Error and Mean Squared Error
- Di erence of the Cumulative Sum
- MSE on the Fast Fourier Transform
- Correlation
- Distance towards the Function Envelope

Mean Absolute Error and Mean Squared Error

The Mean Absolute Error (MAE) and Mean Squared Error (MSE) are common di erence metrics in statistics and machine learning. MSE is less sensitive to smaller errors than MAE. This is of interest because of sensor noise often prevalent in data with a higher sampling rate. Both distance metrics towards the normal $\hat{x}$ with the length n are defined as follows:

$$MAE = \frac{1}{n}\sum_{t=1}^{n}|\hat{x}_t - x_t|, \tag{5.1}$$

$$MSE = \frac{1}{n}\sum_{t=1}^{n}(\hat{x}_t - x_t)^2 \tag{5.2}$$

Di erence of the Cumulative Sum

The di erence of the cumulative sum (in the following MAE Med. Sum, described in [121]) is a distance metric where the signals are cumulatively summed over time before their di erence (in MAE) is computed. Its aim is to be able to take the distance of peaks into account (shown in Figure 5.3), which also makes it more robust towards noise. It is evaluated because its benefit for noisy signals (especially sensor noise) is plausible. It is implemented using the cumsum function in NumPy for a cumulative sum of the sequence[122].

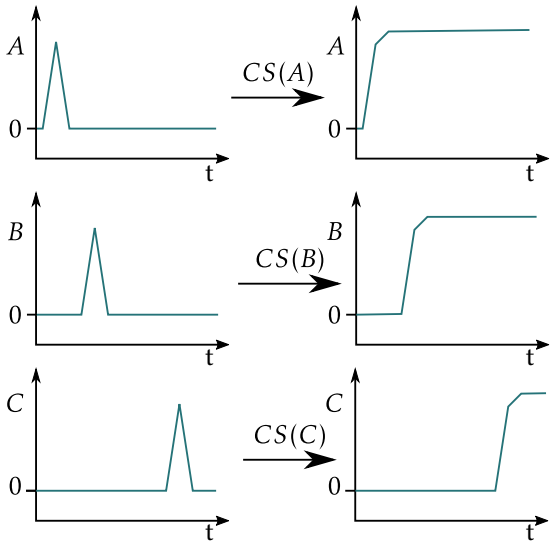


Figure 5.3: Visualization of the cumulative sum (CS) distance metric. A and C are supposed to be closer in distance, which is achieved when first computing the cumulative sum of the time series. Example derived from [121].

MSE on the Fast Fourier Transform

To see if the frequency features of the signals $s(t)$ are influenced by changes in the system, the MSE based on the log-scaled Fast Fourier Transform (FFT) of the signals is also evaluated. The scaling is performed to enhance the higher frequencies of the spectrum. The resulting FFT' is defined as:

$$FFT'(s(t)) = log(|FFT(s(t))|) \tag{5.3}$$

The distance between the FFT' is computed using MSE. It is important to keep in mind that, to process signals like vibration signals accurately, it is important to respect the Shannon-Nyquist sampling theorem. This means that when wanting to analyze a signal of a certain frequency, the sampling frequency must be at least double the signal frequency. Otherwise, features will be lost and skewed due to aliasing.

Cross-correlation

Correlation is a measure of similarity between two vectors. The implementation in the Scipy library [57] of the correlation z between two vectors x and y, with $||x||$ as the length of x and $N = \max(||x||, ||y||)$ is defined as:

$$z[k] = (x * y)(k - N + 1) = \sum_{l=0}^{||x||-1} x_l y^*_{l-k+N-1}. \tag{5.4}$$

This yields an array z with the same length as the inputs. The used value for evaluation in this case is the last element of z, which, if x and y are identical, is the index at which the correlation value is maximal.

Distance towards the Function Envelope

The Hilbert transform yields an upper and lower bound for an oscillating function. This can be used to compute a "tolerance envelope" around a noisy time series to make anomaly detection more robust [123]. To calculate the envelope for a non-oscillating function, first the median-smoothed time series is subtracted from the signals. The average tolerance envelope of all repetitions is computed as the median of all upper bounds and the median of all lower bounds, to yield the average upper and lower limits s_u and s_l, as shown in Figure 5.4. All signal points between these bounds are counted as zero, the remainder is counted in their distance towards the envelope. Therefore, this di erence di (t) of a signal $s(t)$ is computed as:

$$\text{di}_{env}(t) = \begin{cases} s(t) - s_u(t) & \text{if } s(t) > s_u(t), \\ 0 & \text{if} s_l < s(t) < s_u \\ s_l(t) - s(t) & \text{if} s(t) < s_l(t) \end{cases} \tag{5.5}$$

Again, $\text{di}_{env}(t)$ is squared and averaged using the mean to yield the anomaly score per channel. This way, the metric is less sensitive to noisy data, while being sensitive to large outliers from the envelope.

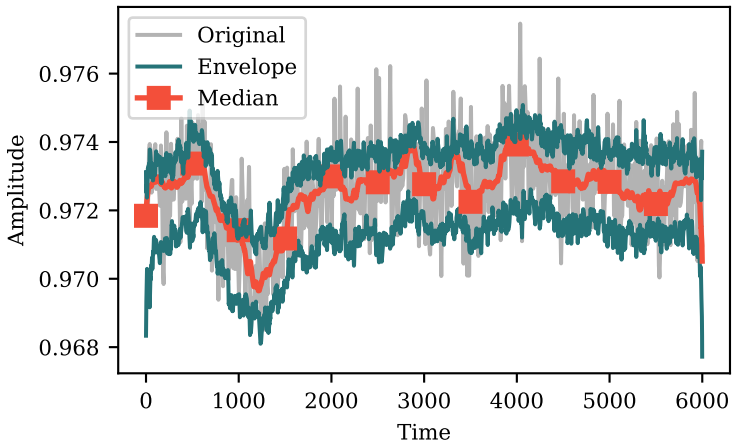


Figure 5.4: Envelope function result using Hilbert function on a sequence that does not oscillate around 0. Before the envelope computation, the windowed median of the signal is subtracted.

5.2.2 Multivariate Outlier Algorithms

Based on the normal, in this case this means the median over time, the distance of each repetition is computed per channel. This distance is then classified. Having shown the di erence metrics, the anomaly classification methods are explained next. They were chosen to work without prior training data. The following anomaly classification methods are used:

- Local Outlier Factor (LOF)
- Modified z-Score

Local outlier factor (LOF)

LOF is a nearest-neighbor based anomaly classification method. Its main parameter is the neighbor count. When one data point is the nearest neighbor of n neighbors, this means that it is classified as normal. If the point is not among its neighbors' nearest neighbors it is classified as an outlier, as shown in Figure 5.5. This means that LOF can be used in a multivariate setting, by computing it based on all distances per signal channel per maneuver.

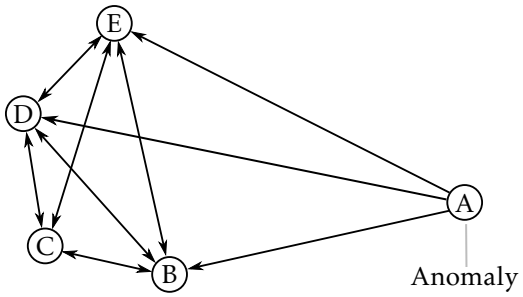


Figure 5.5: Illustration of LOF algorithm with a neighbor-count of 3. Arrows show the nearest neighbors of a node. Point A has several other points in the left cluster as its nearest neighbors, while the nearest neighbors of the points in the left group are in the group. Therefore, point A would be classified as an outlier.

Modified z-Score

The Modified z-Score by Iglewicz and Hoaglin [111] is a statistical score based on the median deviation of a data set. For this problem, it is adjusted for the multivariate case by computing the average Modified z-Score for all channel di erences in a maneuver repetition. If the average z-Score is above the set threshold value (3.5), the complete maneuver counts as an anomaly. The modified z-Score by [111] is defined as:

$$score = |0.6745(d_i - \tilde{d})/MAD(d)|, \tag{5.6}$$

In this chapter, the Modified z-Score for a maneuver repetition with c channels based on the di erences d of the repetition is defined as:

$$score = \frac{1}{c}\sum_{i=1}^{c}|0.6745(d_i - \tilde{d})/\mathrm{MAD}(d)|, \tag{5.7}$$

with

$$MAD(d) = median(|d_i - \tilde{d}|) \tag{5.8}$$

with $\tilde{d}$ as median value over time of all repetitions and MAD as the median absolute deviation. The threshold is chosen according to the

original authors, who equate a Modified z-Score above 3.5 to an anomaly. Though this in not described by [111], if MAD(d) equals 0, more than half of all samples are equal to $\tilde{d}$. This means that in this case, samples d_i that are unequal $\tilde{d}$ should also be classified as anomalous.

5.2.3 Interpretation of Results

Since the outlier classification is based on the di erence between the signal types of a maneuver repetition, it is possible to visualize the result of the anomaly detection for further inspection. Figure 5.6 demonstrates this. Plotting the di erence metrics with one line per maneuver in parallel coordinates shows the channels with the most deviation from the normal. This can help technicians and engineers find the channels with the most influence on the classification decision, and support them in a faster verification.

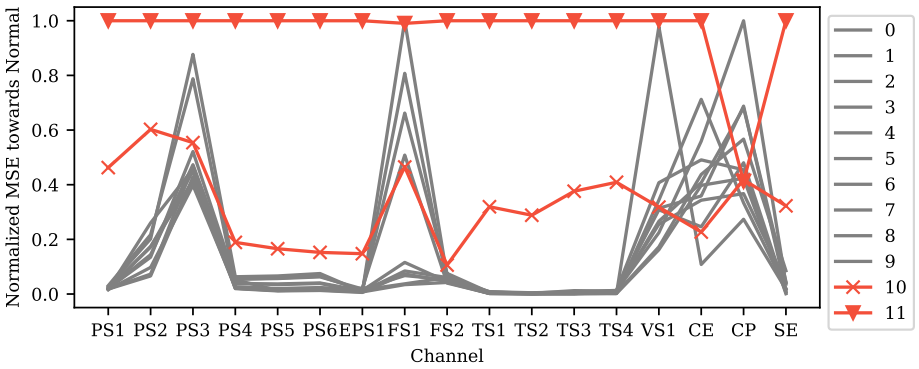


Figure 5.6: Visualization of a classification result with parallel coordinates. Each line symbolizes one measurement. For each channel, the anomaly score is shown. Repetitions 10 and 11 are marked as classified as anomalous. This way, the relevant channels can be selected for further investigation.

5.2.4 Real-Time Anomaly Detection and Concept Drifts

The shown methods can be used for real time anomaly detection by comparing the output of one tested system to all others performing the same maneuver at the same progress in the test run in a sliding window. It is especially important to reset the window for changing operating points, e.g., in temperature modulated test benches. Once a significant temperature change is performed, old measurements can cause false positive anomaly classifications. Therefore, the maneuver repetitions for comparison must be recorded at the same temperature. As a rule of thumb, the experiments have shown that eight to ten repetitions are sufficient, though less might work as well given that only a minority of parts fail at a time. This also works for the anomaly detection under concept drift. Since there is no explicit model trained, it is possible to detect anomalies even under system changes.

5.3 Experiments

The di erence computations and subsequent anomaly classification methods are examined with regards to the dataset in the experiments described in this section. They are structured in three parts

1. All distance metrics from 5.2.1 for each part (cooler, valve, pump, hydraulic accumulator) classified with the Modified z-Score
2. All distance metrics with di erent anomaly detection algorithms for each of the sub-components
3. Test of classification methods with the MSE metric under concept drift over the system

The accuracy in all experiments is evaluated in 10-fold cross validation using the area under the ROC curve statistic in Scikit-learn. During all experiments, a randomized distance function is also computed as a baseline metric.

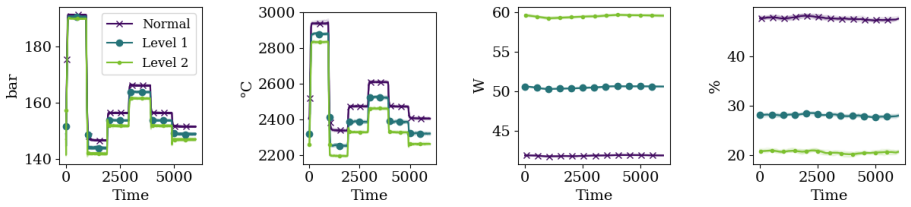
5.3.1 Hydraulic Test Bench Data Set

The hydraulic test bench data set used consists of labeled multivariate measurements of a hydraulic system under test. The system has four di erent components (cooler, valve, pump, hydraulic accumulator) at three to four discrete levels of wear, with ten repetitions of each combination. It consists of 17 di erent signal channels (among others six pressure, two volume flow and four temperature) with di erent sampling rates. In the following all channels are re-sampled to the highest sampling rate (100 Hz) using linear interpolation and rescaled to an amplitude of $[0, 1]$. Not every type of anomaly is visible in all channels equally, as shown in Figure 5.7-5.10. The degradation of the cooler is clearly visible across several channels, while the deteriorating valve is mostly visible in the "Temp. 1" channel, and not clearly discernible in the other channels.

For the simulation of the concept drift, a data set was constructed by taking di erent component wear levels in a continuously degrading order, starting with the normal for all components. This will be described in detail later.

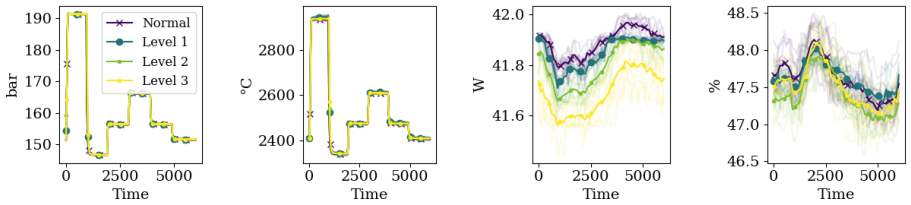
5.3.2 Per Channel Anomaly Detection

To evaluate distance measures, the Modified z-Score is used in a semi-supervised setting per measurement channel. The mean and MAD of the normal state was modeled using 8 samples from the normal data set, where every sub-component is in best condition. The testing was done using 2 from the normal, and 20 abnormal repetitions (test set). The distance measures were computed per signal channel. Measurements with a Mod. z-Score above 3.5 were classified as anomalies. Results are shown in Tables 5.3a - 5.3d for wear on the cooler, valve, pump, and hydraulic accumulator. In these experiments, the detection accuracy for wear in the cooler is better than for the valve. This can be explained by the data set plots shown before



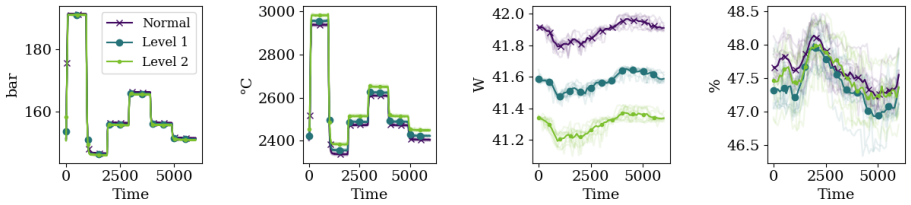
(a) Pressure 1 (b) Motor Power (c) Temp. 1 (d) Cooling Effic.

Figure 5.7: Di erent levels of cooler degradation. The median measurement is visualized in opaque with line-markers. The di erences in amplitude are clearly visible in all channels.



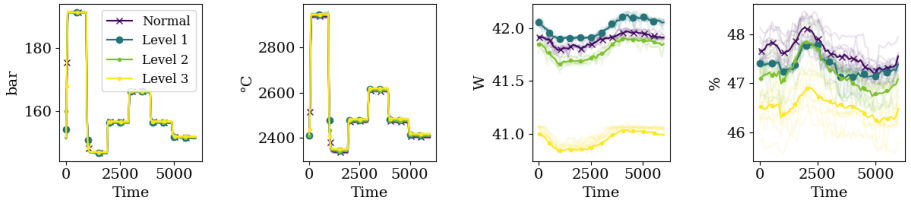
(a) Pressure 1 (b) Motor Power (c) Temp. 1 (d) Cooling Effic.

Figure 5.8: Di erent levels of valve degradation. The di erence in amplitude is not as clearly visible as in Figure 5.7, especially for Temp. 1 and Cooling Efficiency.



(a) Pressure 1 (b) Motor Power (c) Temp. 1 (d) Cooling Effic.

Figure 5.9: Changes in signals for degradation of the pump. The motor power and temperature changes are clearly visible.



(a) Pressure 1 (b) Motor Power (c) Temp. 1 (d) Cooling Effic.

Figure 5.10: Changes for the hydraulic accumulator. The cooling efficiency and temperature show di erences in amplitude.

(Figure 5.7 and 5.8), where the cooler wear was detectable easily for the human eye. Changes in the cooler, for example, were the most difficult to detect in the signal channel 7. Figure 5.1 shows that the general accuracy of the distance metrics per part of the Envelope-Di , MSE, MAE and MAE Med. Sum is similar, with Envelope and MSE being the best. The MSE-FFT performs worst of all, followed by correlation.

Table 5.1: Experiments Part 1: Comparison with classification accuracy of all distance metrics over all parts using Mod. z-Score

	Cooler	Valve	Pump	Hydro	All
MAE to Median	0.90	0.67	0.86	0.71	0.78
MSE to Median	0.91	0.73	0.88	0.73	0.81
MAE Med. Sum	0.97	0.62	0.90	0.64	0.78
MSE to FFT	0.61	0.59	0.56	0.56	0.58
Correlation	0.92	0.55	0.74	0.58	0.70
Envelope	0.89	0.78	0.87	0.76	0.82
Random	0.50	0.50	0.50	0.50	0.50
All	0.81	0.63	0.76	0.64	0.71

5.3.3 Di erent Outlier Classification Methods

The outlier classification methods were then evaluated in a semi-supervised, multivariate setting, to evaluate their accuracy for the wear of single components. This means that the algorithm's normal mode is extracted based on 8 samples of the normal data set, and is then evaluated by classifying anomalies from 2 remaining normal and 20 abnormal samples from each wear level per part. Based on this, the Local Outlier Factor with n=5 and the Modified z-Score with a standard threshold of 3.5 were evaluated. Figure 5.4 shows the results: In this use case, LOF works the best in combination with MAE. Notice how this is di erent from the channel-wise anomaly detection case, where Envelope-Di and MSE performed best.

5.3.4 Concept Drift Analysis

For the concept drift evaluation using unsupervised anomaly detection, a simulated aging process was constructed by increasing change in the system with time series data at ten di erent time steps. For this, a system life cycle from intact to maximum wear of all sub-components was simulated. The taken measurements can be seen in Table 5.5. The normal wear over time was adjusted by replacing the corresponding column with the sub-components normal and abnormal. For example, at $t = 3$, the accuracy for Cooler anomalies were assessed with 10 normal samples of $C = 50\%$, $V = 67\%$, $P = 100\%$ and $H = 100\%$ combined with 2 abnormal measurements with $C = 0\%$. This way, a comparable evaluation was achieved for all sub-components.

In general, for this use case, the Modified z-Score in combination with the MSE performed the best, as shown in Table 5.6. Table 5.8a and 5.8b

Table 5.2: Experiments 1: Comparison of classification accuracy with distance metrics per channels with of Mod. z-Score.

(a) Cooler

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
MAE to Median	0.50	0.80	0.55	1.00	1.00	1.00	0.97	0.94	0.97	0.97	0.90	0.95	0.97	0.90	1.00	0.85	0.97
MSE to Median	0.82	0.97	0.85	1.00	1.00	1.00	0.95	0.68	1.00	0.95	0.88	0.93	0.95	0.93	1.00	0.82	0.79
MAE Med. Sum	1.00	1.00	1.00	1.00	1.00	1.00	0.97	0.79	1.00	1.00	0.90	1.00	1.00	1.00	1.00	1.00	0.83
MSE to FFT	0.68	0.82	0.66	0.82	0.57	0.50	0.67	0.88	0.58	0.50	0.72	0.41	0.53	0.53	0.36	0.51	0.55
Correlation	1.00	0.95	0.63	1.00	0.95	1.00	1.00	0.75	1.00	0.88	0.85	0.95	0.93	0.88	0.97	0.93	1.00
Envelope	0.85	0.88	0.88	0.95	0.97	0.97	0.95	0.69	0.97	0.93	0.88	0.95	0.93	0.93	0.97	0.85	0.62
Random	0.51	0.54	0.50	0.47	0.48	0.50	0.50	0.46	0.51	0.52	0.47	0.47	0.50	0.51	0.48	0.52	0.50

(b) Valve

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
MAE to Median	0.50	0.80	0.55	0.51	0.50	0.50	0.68	0.94	0.57	0.91	0.76	0.86	0.84	0.46	0.56		
MSE to Median	0.82	0.97	0.85	0.52	0.54	0.51	0.92	0.85	0.59	0.92	0.86	0.84	0.87	0.48			
MAE Med. Sum	1.00	1.00	0.54	0.61	0.58	0.56	0.54	0.50	0.50	0.73	0.77	0.60	0.51	0.50			
MSE to FFT	0.68	0.82	0.82	0.47	0.52	0.45	0.68	0.83	0.53	0.54	0.66	0.37	0.53	0.55			
Correlation	0.51	0.46	0.50	0.51	0.46	0.50	0.50	0.50	0.57	0.84	0.62	0.85	0.55	0.50	0.49	0.50	0.51
Envelope	0.85	0.88	0.88	0.79	0.81	0.82	0.92	0.80	0.71	0.92	0.87	0.87	0.85	0.48	0.57	0.63	0.62
Random	0.48	0.49	0.50	0.47	0.52	0.51	0.44	0.48	0.50	0.51	0.50	0.50	0.48	0.47	0.54	0.45	0.52

(c) Pump

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
MAE to Median	0.50	0.80	0.55	1.00	1.00	0.99	0.97	0.95	0.97	0.97	0.90	0.95	0.97	0.70	0.65	0.69	0.97
MSE to Median	0.82	0.96	0.84	1.00	1.00	1.00	0.95	0.75	1.00	0.95	0.88	0.93	0.95	0.74	0.69	0.68	0.82
MAE Med. Sum	1.00	1.00	1.00	1.00	1.00	1.00	0.94	1.00	0.89	1.00	0.90	1.00	1.00	0.56	0.50	0.58	0.97
MSE to FFT	0.60	0.70	0.56	0.68	0.57	0.49	0.50	0.56	0.61	0.55	0.72	0.39	0.61	0.50	0.43	0.47	0.51
Correlation	0.96	0.57	0.50	0.73	0.69	0.66	0.89	0.50	0.75	0.88	0.85	0.95	0.93	0.50	0.57	0.62	1.00
Envelope	0.85	0.88	0.86	0.95	0.97	0.97	0.95	0.67	0.97	0.93	0.88	0.95	0.93	0.74	0.70	0.72	0.88
Random	0.48	0.50	0.53	0.51	0.49	0.49	0.50	0.48	0.42	0.50	0.50	0.48	0.48	0.51	0.50	0.49	0.50

(d) Hydraulic Accumulator

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
MAE to Median	0.50	0.79	0.53	0.61	0.50	0.50	0.72	0.72	0.88	0.85	0.90	0.93	0.76	0.70	0.71	0.61	0.81
MSE to Median	0.82	0.79	0.70	0.63	0.60	0.56	0.79	0.55	0.90	0.84	0.88	0.90	0.75	0.75	0.74	0.58	0.59
MAE Med. Sum	0.62	0.56	0.50	0.76	0.69	0.65	0.52	0.50	0.77	0.75	0.90	0.75	0.69	0.66	0.51	0.51	0.51
MSE to FFT	0.60	0.76	0.71	0.53	0.55	0.50	0.64	0.65	0.52	0.48	0.74	0.36	0.51	0.48	0.40	0.57	0.51
Correlation	0.50	0.46	0.50	0.60	0.46	0.50	0.50	0.50	0.58	0.78	0.78	0.95	0.65	0.50	0.57	0.46	0.54
Envelope	0.85	0.88	0.68	0.80	0.73	0.73	0.78	0.53	0.91	0.83	0.88	0.93	0.73	0.73	0.71	0.62	0.54
Random	0.53	0.50	0.50	0.51	0.55	0.51	0.51	0.51	0.52	0.51	0.52	0.50	0.50	0.50	0.49	0.49	0.48

Table 5.4: Experiments Part 2: Comparison of distance metrics in combination with the two di erent anomaly detection algorithms for the multivariate case. Based on the best metric (MSE), the Modified z-Score has the best accuracy.

	LOF	z-Score	All
MAE to Median	0.92	0.93	0.93
MSE to Median	0.97	0.97	0.97
MAE Med. Sum	0.86	0.69	0.78
MSE to FFT	0.40	0.48	0.44
Correlation	0.75	0.66	0.70
Envelope	0.95	0.95	0.95
Random	0.50	0.51	0.50
All	0.76	0.74	0.75

show the di erent accuracies per component over time for this constellation. It is noticeable that the accuracies for all parts except the cooler decreases with $t \geq 6$. Nonetheless, this experiment is a proof of concept that, up to a certain degree, accurate anomaly detection under concept drift is possible using this method. Changes in the cooler are detectable with the highest accuracy. The model's accuracy over time degrades after $t = 6$, which coincides with the maximum wear level of the cooler.

Table 5.5: Test bench measurements taken for the simulation of the concept drift dataset. To create the data set for each subcomponent, the normal trend over time was used and the respective column was replaced with the components' normal and abnormal columns to create the specific dataset. The percentages stem from the discrete wear levels in the data set.

	Normal Integrity over Time				Cooler Test		Valve Test		Pump Test		Hydr. Test	
t	Cooler	Valve	Pump	Hydr.	Normal	Anomaly	Normal	Anomaly	Normal	Anomaly	Normal	Anomaly
0	100%	100%	100%	100%	100%	50%	100%	67%	100%	50%	100%	67%
1	50%	100%	100%	100%	50%	0%	100%	67%	100%	50%	100%	67%
2	50%	67%	100%	100%	50%	0%	67%	33%	100%	50%	100%	67%
3	50%	67%	100%	100%	50%	0%	67%	33%	100%	50%	100%	67%
4	50%	67%	50%	67%	50%	0%	67%	33%	50%	0%	100%	67%
5	50%	33%	50%	67%	50%	0%	67%	33%	50%	0%	67%	33%
6	0%	33%	50%	67%	50%	0%	67%	33%	50%	0%	67%	33%
7	0%	0%	50%	67%	50%	0%	33%	0%	50%	0%	67%	33%
8	0%	0%	0%	33%	50%	0%	33%	0%	50%	0%	33%	0%
9	0%	0%	0%	0%	50%	0%	33%	0%	50%	0%	33%	0%

Table 5.6: Experiments 3: Comparison of all di erence scores and outlier algorithms for concept drift

	LOF	z-Score	All
MAE to Median	0.85	0.93	0.89
MSE to Median	0.88	0.95	0.91
MAE Med. Sum	0.88	0.71	0.80
MSE to FFT	0.53	0.61	0.57
Correlation	0.81	0.74	0.77
Envelope	0.88	0.93	0.91
Random	0.50	0.50	0.50
All	0.76	0.77	0.76

5.4 Chapter Summary

In this chapter, a framework for unsupervised test bench anomaly detection was proposed and evaluated on the example of one data set of a hydraulic system, without and with a simulated concept drift. For this, several time series distance computation methods and two anomaly classification algorithms were examined and their limitations were shown.

There are several advantages of this method. It can be employed from the start of a test bench and with a small sample size. It is compu-

Table 5.7: Experiments Part 3: Results of experiments concerning the concept drift of the system

(a) Local Outlier Factor

	0	1	2	3	4	5	6	7	8	9
Cooler	1.00	0.99	0.95	0.95	0.98	0.99	0.90	0.93	0.97	0.99
Valve	0.99	1.00	0.91	0.91	0.96	0.99	0.78	0.49	0.90	0.88
Pump	0.99	0.99	0.93	0.93	0.91	0.99	0.93	0.47	0.47	0.94
Hydro	0.78	0.99	0.93	0.93	0.97	0.95	0.42	0.90	0.69	0.56
All	0.94	0.99	0.93	0.93	0.96	0.98	0.76	0.70	0.76	0.84

(b) Modified z-Score

	t= 0	1	2	3	4	5	6	7	8	9
Cooler	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Valve	1.00	1.00	1.00	1.00	1.00	1.00	0.94	0.60	0.94	0.94
Pump	1.00	1.00	1.00	1.00	1.00	1.00	0.94	0.78	0.81	0.94
Hydro	0.93	1.00	1.00	1.00	1.00	0.99	0.78	0.96	0.74	0.56
All	0.98	1.00	1.00	1.00	1.00	1.00	0.91	0.84	0.87	0.86

tationally efficient and can be re-performed frequently, making it robust when dealing with changing operating points of a system that is aging under load. Furthermore, the classifier inputs can be visualized to show the channels with the most deviation from the normal, which makes this step more easily interpretable for experts.

The results demonstrated that the choice of distance metric had a larger influence on the prediction accuracy than the following classifier. For the given dataset, the MSE of the signals in the time domain performed best, while the distance of the FFT of the signals performed worst. This means that this method is applicable for anomaly detection in the time domain, but – at least for the given dataset – not in the frequency domain.

From this work, there are several areas of future research. This chapter focuses on one data set that was taken from one hydraulic system. The great benefit of this data set were the di erent wear levels in the sub-components, the diverse measurement signals, and the accurate labeling. For future work, it would be of interest to evaluate even more datasets and sensor types, which would be beneficial for showing the statistical significance of the presented method, especially in the concept drift analysis.

For now, only one distance measure was used as input for the classifier. A combination of multiple feature types, such as MSE and the correlation of measurements might make predictions more accurate. Using the presented methods in an ensemble training approach would also be of interest. Additionally, evaluating these methods on more systems and use cases is required.

This chapter has shown that the proposed method was the least e ective for anomaly detection in the frequency domain. Such data has significant importance for topics like vibration analysis, which is needed for the anomaly detection in rotating components (e.g., ball bearings [124])

or for imbalance detection. The goal of this thesis is to examine di erent uses of data science for the anomaly detection in test benches. Therefore, methods for data in the frequency domain will be focused on in the next chapter.

6 Data-based Supervised Anomaly Classification for Vibration Data*

The goal of the prior chapter was to find a method for unsupervised anomaly detection that would be easily understandable for humans. It was shown that, for the evaluated data set, anomaly classification based on a difference metric between two Fourier transformed signals was not as effective as applying the same method to the signal in the time domain.

There are failure modes in a test bench, though, where the relevant data features are better identifiable in the frequency domain (see Section 2.2.2 and 2.2.3), particularly for systems with rotating and thereby moving components. Examples for this are failures of ball bearings [124] or gear boxes [125]. Such defects can cause other components to fail faster due to the increased vibrations in the system [11]–[13].

With the rise of deep learning, deep neural networks (DNNs) are also being used for vibrational data analysis [126]–[131]. One benefit of DNNs is, that network layers, for example convolutional layers, are used to automatically extract relevant features from data and perform filtering implicitly based on a given task [132]. While these networks show high classification accuracy, no research could be identified that evaluates the performance of CNNs on data in the time domain vs. the frequency domain. Janssens et al., e.g., compare pre-extracted features fed into non-neural network models with the performance of a CNN on frequency domain data [132] for fault detection in rotating machinery. They do not check the accuracy change before and after the frequency transformation.

Therefore, the focus of this chapter will be the investigation on how much DNNs profit from frequency-transformed features. Due to the given labels in the data, this will be approached in a supervised way, but it can be assumed that the following results can also benefit unsupervised ML tasks.

Time series recordings of sensor acceleration attached to a system, known as vibrational data, can provide valuable insights into the condition of the system. [132]–[134]. Renwick and Babson [135] describe how anomalies can cause changes in vibrational amplitude, frequency, phase, and modulation. Amplitude shows the amount of change, frequency the repetition rate of damage, phase is the time-offset between signals, and modulation means that lower frequency vibration from a damage can excite higher frequency vibrations in a system due to resonances (cf. [135], p.325). The transformation of vibrational data to the frequency domain is a common preprocessing method to retrieve more information on the

*Parts of this chapter have been published in "O. Mey and D. Neufeld, "Explainable AI Algorithms for Vibration Data-Based Fault Detection: Use Case-Adadpted Methods and Critical Evaluation", *Sensors*, vol. 22, MDPI, Ed., pp. 1–22, 2022, Source Code link: https://github.com/o-mey/xai-vibration-fault-detection. DOI: 10.3390/s22239037"

system. Isolating relevant frequencies in the vibrational data per manual feature engineering can pose challenges [136]. One reason is the change of vibration signature depending on operating mode. This stems, among others, from the system's own resonance frequencies. Naturally, the vibrational signal, especially with regards to the current operating mode, is di erent for every system, and every moving system causes noise and vibration, even in normal conditions [137]. An additional limitation is that there can be di erences from one system to another, structurally identical one, stemming from the distribution of fabrication parameters such as the force to tighten screws [138] or radii of ball bearings.

Data pre-processing techniques are implemented to extract information on the frequency-dependent aspects of vibrational signals. One such technique is the Fourier transform, which converts the data from the time domain to the frequency domain. To get a better resolution of the frequencies at a certain point in time, short-time Fourier transforms (STFT) is an option [126]. This means that the signal is divided into smaller segments before the application of the Fourier transform to each segment. The result of the STFT is 2D datasets where one axis corresponds to the time and the other axis to the frequencies in the signal. If vibrational data is recorded in a RPM (revolutions per minute) ramp, i.e., if the system rotating speed is increased while recording, other transforms known as RPM maps can be used, which also yield 2D data. In frequency-RPM maps, the x-axis corresponds to the frequencies in the Fourier-transform of the signal, while the y-axis relates to the system RPM. Parts of the spectra that relate to multiples of the rotating frequency of the system will appear in sloped, straight lines, and their location in the frequency axis will increase with raising RPM [139], [140].

The second variant of transformation evaluated in this chapter are order-RPM maps, where the frequency axis of the frequency-RPM map is normalized based on the systems current rotational frequency. This means that at order = 1 the main excitation of the system is visible. An advantage of this is that vibrations from elements like ball bearings, which cause excitations at a multiple of the systems frequency to appear, are expected to cause straight lines parallel to the 1st order in the order spectrum. System resonances result in sloped lines [141], [142].

Both transforms can be leveraged by experts for feature engineering and system analysis. The assessment of vibration data demands in depth background knowledge on the system itself, as well as experience in signal processing. Still, there is the question whether CNNs also benefit from such preprocessing.

The goal of this chapter is therefore to investigate the influence of data transformation on the performance of deep learning models based on a real-world dataset by Mey at al. [143] of a rotating motor with an imbalance. The same CNN neural network architecture will be trained for the same data after di erent data transformations: Min-Max scaled time series, frequency-RPM maps, and order-RPM maps.

While the state of the art also o ers other data preprocessing methods (e.g., the wavelet transforms), the goal is to examine whether there

are di erences in the prediction accuracy of di erent transformations and to show that there are benefits to comparing methods. Additionally, with the methods shown in this chapter it is not possible to say whether the CNNs take the correct features (i.e., frequencies or orders) of the data into account, since these models by themselves are not interpretable to humans. The application of explainable AI (XAI) for this problem is the focus of the next chapter.

The next sections are structured as follows: First, related work on anomaly detection in combination with vibration analysis is presented. Then the datasets used in this and the next chapter along with transformations will be described. In the experiments section, the models' results will be compared along with their performance on the testing dataset.

6.1 Related Work

First, an overview will be given on current predictive monitoring approaches, followed by ML models used in them and di erent preprocessing steps in related work.

Vibrational analysis is an important part of Predictive Maintenance (PM) and Condition Monitoring (CM) [11], [128], [135]. Rotating objects, under surveillance are for example wind turbines [27], [144], rotors [145], rolling element ball bearings [124], [146], [147], gear boxes [27], [125], [148], drive trains [138] and rotating shafts [129].

There are di erent approaches for the PM and CM models, ranging from manual feature engineering to fully automated, ML based methods. The ML based models are used to, to an extent, extract relevant features automatically. Di erent models have been used for vibration-based anomaly detection [13], [124], [127], [145], [149]. Examples are K-Nearest Neighbors [144], [148], [150], K-Means [144], and Support Vector Machines (SVM) [13], [144], one class SVM (OCSVM) [136], Deep Belief Networks [146], Random Forest [129], and Naive Bayes Classifiers [148]. In the domain of Deep Learning [128], autoencoders [146], CNNs [132], [151] and recurrent networks such as LSTM [12] are used for the anomaly detection in PM.

Regarding the domain of vibration analysis, di erent approaches for the preprocessing and feature extraction for anomaly detection and classification were used in related work. In their review on vibration analysis techniques for rotating machines, Singh and Vishwakarma [137] distinguish between the data in time domain, frequency domain and time-frequency domain. Signal envelopes in the time and frequency domain are further examples [124]. Going from a statistical approach, metrics such as the root mean square, the mean, variance, skewness, kurtosis and the crest factor of the raw signal [124], [137], [152] are common. In the time-frequency domain wavelet-transform, frequency spectra [124], [132], [153], order spectra [154] are used.

This chapter focusses on the benefits of frequency- and order- RPM maps compared to data in the time domain for the performance of CNN classifier models. The imbalance data set was used as input for classifier

models with Random Forests and Hidden Markov Models in prior research [129] using the frequency spectrum of the signal.

6.2 Methods

The methodology of this chapter is structured in the dataset used for evaluation, the preprocessing of the data and the structure of the classifier CNNs.

6.2.1 Frequency Domain Preprocessing

The resolution of the data in the time-frequency domain was chosen to produce visually satisfactory resolution for the frequency- and order-RPM maps. The sequence length and feature count of both preprocessing methods are similar. The time series data was split in lengths the same as the number of features in the frequency domain. This results in less samples in the order domain than the frequency domain, and less samples in frequency than the time domain. To keep the number of iterations per epoch the same and make training comparable, a random subset of time and order data was used.

Figure 6.3 shows the dataset in the time domain and its frequency RPM maps and the order-RPM maps. In the time domain, the amplitude of the signal oscillations is visibly higher with imbalance. The mean of the signal remains zero. In the frequency-RPM maps, this change in amplitude is also visible based on the intensity values of the pixels. It is also noticeable that the general intensity of vibration increases with higher RPM of the system. Lines in the spectrum which are parallel to the y-axis of the system, stem from the system's resonance frequencies independent of the rotation speed. The diagonal lines in the spectrum indicate vibration frequencies which increase with rising motor RPM. In the order-RPM map, the change of amplitude with rising RPM is not as obvious as in the frequency-RPM map, but still noticeable. Here, the major changes occur at the order of 1, which is the main vibration frequency of the system. There are also spectral lines parallel to the 1st order in the range of orders 17 to 20. It can be assumed that these come directly from the rotation of the motor and are excited by ball bearing guiding the shaft of the system.

6.2.2 CNN Model

For the classification, a CNN-based neural network model adapted from [129] is used. The configuration is shown in Figure 6.1. Fully convolutional layers followed by max-pooling layers were used to reduce the size of the features. The final activation function of the model is softmax, for multiclass classification with two classes. Batch norm [155] layers were used to improve stability during training and better model performance. For all kinds of preprocessing, the same model structure was used.

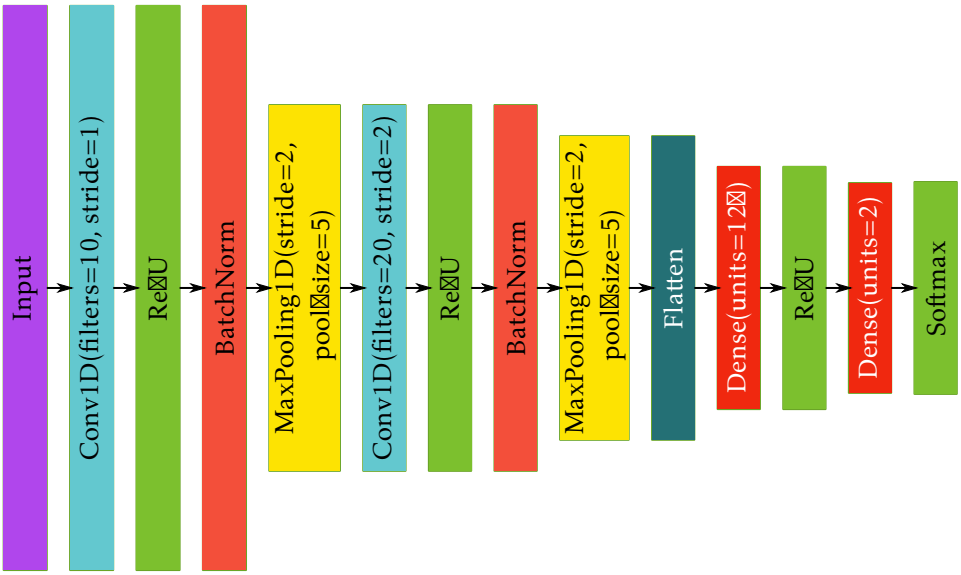


Figure 6.1: CNN model used for classification.

6.3 Experiments

A more detailed description of the dataset for the experiments and the setup of the training and evaluation are described next.

6.3.1 Dataset

The data published by Mey et al. in [129] is a vibrational dataset of a rotating motor system with a load attached to the end of the shaft. The motor rotates at different RPMs. The load contains an imbalance of varying levels, from none to a maximum value, in five steps. The dataset is shown in Figure 6.3.

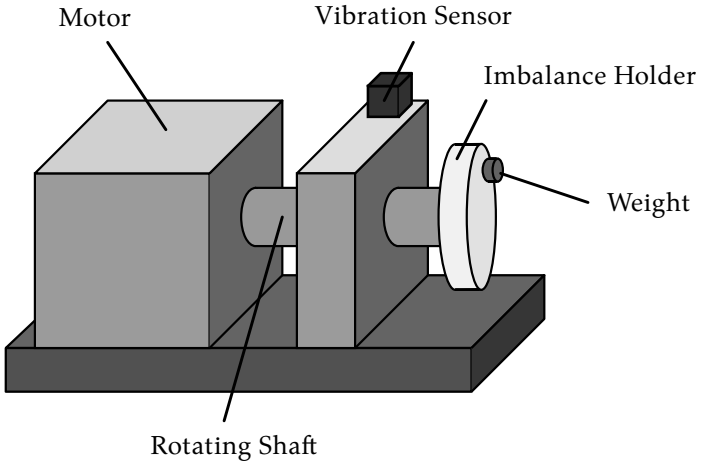


Figure 6.2: Schematic of the setup of the data recorded by [143] and used in this chapter (based on Figure 1 in [156].)

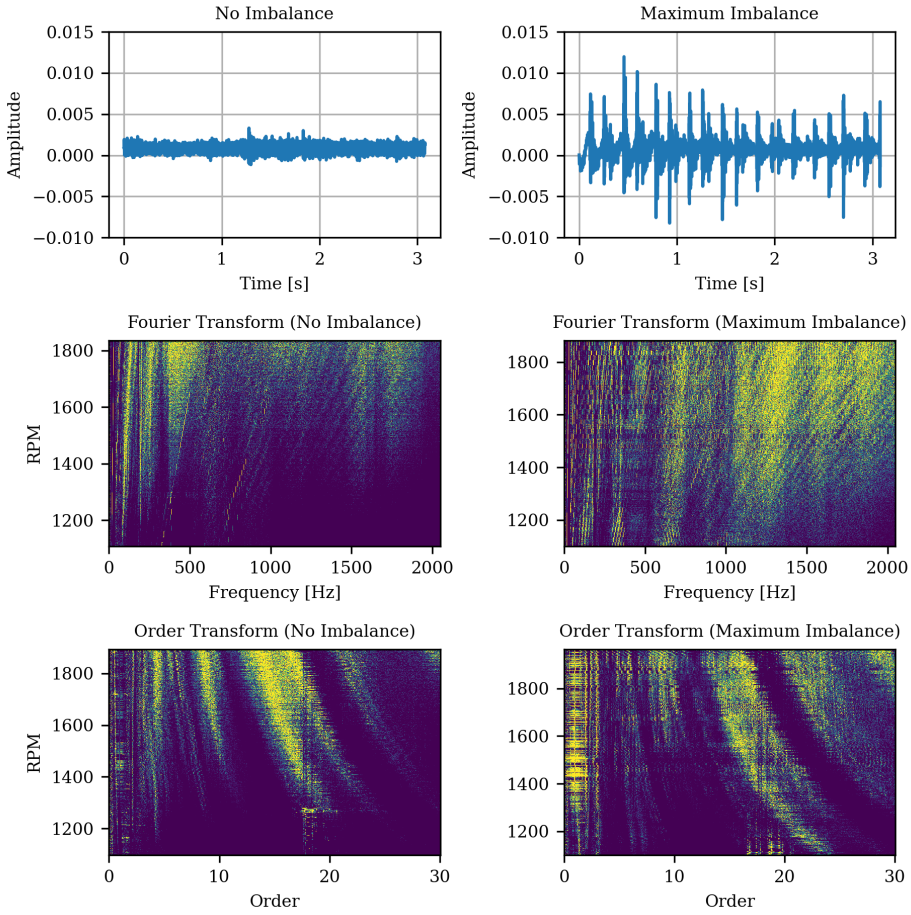


Figure 6.3: Vibrational dataset in time domain and after preprocessing into frequency- RPM maps and order-RPM maps.

6.3.2 Results

The same training procedure was followed for all runs. The models were trained for 150 epochs each using the Adam optimizer. The validation loss was recorded during training to avoid overfitting, and the best model weights on the validation set were used for prediction on the test dataset.

Figures 6.4 shows the trends of losses for the di erent preprocessing methods. The classification accuracies and losses for the testing data set are shown in Table 6.1.

There is a noticeable di erence in training between the three methods. The model trained on the time-domain data converges at a higher final loss compared to frequency- and order maps. During training, the order map results in a lower loss than the frequency map, but for the testing data set, the FFT results in a slightly higher accuracy. This may be happening due to overfitting. The creators of the data set claimed to have taken apart the system and rebuilt it prior to recording the testing data set, which can change the system's properties. This is a probable cause to the changes in accuracy between training and testing.

To summarize the results, it can be said that even though CNNs are able to perform filtering of input data due to the way they are designed, applying preprocessing prior to training can improve prediction accuracy.

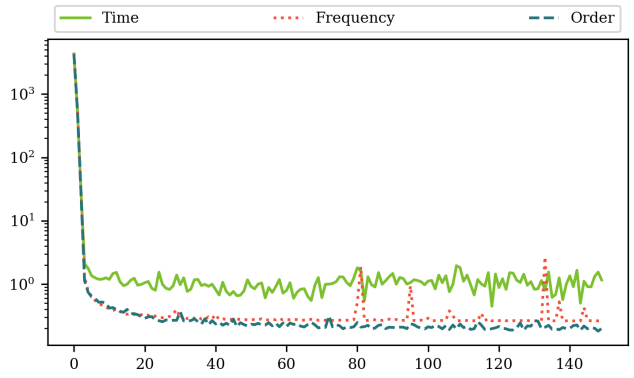


Figure 6.4: Loss during training on the validation data. The model converges earlier to a worse result for the data in the time domain than for the preprocessed data.

Table 6.1: Evaluation scores for the test data set based on di erent kinds of data preprocessing

	Accuracy			Loss		
Domain	Time	Frequency	Order	Time	Frequency	Order
Score	0.877	**0.996**	0.976	2.67	**0.92**	1.28

6.4 Chapter Summary

It was shown that the pre-processing of data has an influence on the prediction accuracy of convolutional neural networks. The accuracy improved notably when using frequency and order transformed signals instead of time series for the complex data set from a real-life mechanical system.

Further possibilities for evaluation would be the comparison of more systems and more diverse kinds of failures (e.g., gear boxes, ball bearings), to see if generalized rules for choosing frequency-RPM map or the order-RPM maps can be derived. Given the results, an expert would need further information on the reasoning of the model. For example, it is necessary to know, which frequencies and orders were relevant to the CNNs prediction. This serves as a motivation for the methods that will be developed in the next chapter.

7 Explaining CNNs for Classification of Vibrational Data*

The work described in this chapter aims to make vibrational classification models of data in the time-frequency domain understandable for system experts. This is relevant for the context of a test bench, but even more for safety critical domains, such as wind turbines [144] or plane rotors [145]). Here, it is essential that a ML model is interpretable to increase acceptance and decrease risk of false predictions. If decision makers, such as engineers or technicians, do not understand how a model reached a certain result, they might trust a model's output more than appropriate or mistrust it. In the latter case, the end users could ignore the model or overwrite its outputs.

In the last chapter, the benefits of frequency-based preprocessing of data for the classification of vibrational data using CNNs was examined, with a focus on the time-frequency domain. The models' accuracies improved with the preprocessing, making this a useful support for the DL-based vibrational anomaly detection. Based on this, an open question is how the expert user can benefit from this method. Since DNNs are black box models [157], it was not possible to derive relevant frequencies or orders of the data to the models' outputs just based on the CNNs parameters after training. So, while a model's accuracy on a data set is satisfactory, the question remains whether the DNN took the correct subset of input features into account. This can lead experts to doubt or trust the output of the model more than technically appropriate. Furthermore, understandable models can be used to pre-filter and reduce the number of input features, to make the model more efficient and robust. This process is shown in Figure 7.1 using the imbalance data set from the last chapter as example: the time series data is preprocessed, in this case using frequency-RPM maps and order-RPM maps, and a classifier model is trained to predict imbalance. Based on this, XAI methods are used to highlight relevant parts of the data for humans, who in this case are the engineers and technicians.

There are several pieces of related work on using saliency methods for rotating system data. Commonly, Layer-wise Relevance Propagation (LRP) [158] or Class-activation maps (CAMs) methods [159], [160] are used. While these methods are reported to provide satisfactory results, they are not optimized towards the properties of the spectral data prevalent in vibrational analysis, namely that certain frequencies or orders can be associated with certain defect modes in a system. Instead, these methods

*Parts of this chapter have been published in "O. Mey and D. Neufeld, "Explainable AI Algorithms for Vibration Data-Based Fault Detection: Use Case-Adadpted Methods and Critical Evaluation", *Sensors*, vol. 22, MDPI, Ed., pp. 1–22, 2022, Source Code link: https://github.com/o-mey/xai-vibration-fault-detection. DOI: 10.3390/s22239037"

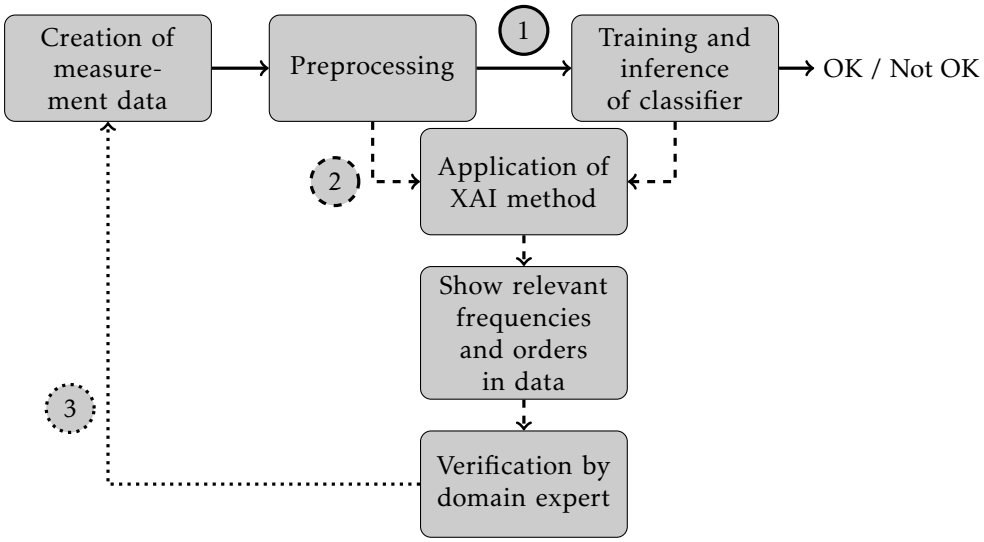


Figure 7.1: Flow of information through training until explanation. In step (1), the data is preprocessed, and a classifier is trained to classify the state of the system. Next, in (2), an XAI method is used to explain the features of the data relevant for the classifier during inference. Finally, an expert uses the XAI results to verify the model (3), and if necessary, adjust the data creation process and the preprocessing of the data.

use the spectral transformed data as images, applied in the same way as for conventional image processing.

Therefore, this chapter presents a new perturbation-based method coined Spectral LIME based on Local Interpretable Model-Agnostic Explanations (LIME) by [161], for frequency- and order-RPM maps of vibrational data, which aims to create a more domain specific explanation of the results. This method will be compared to the standard LIME implementation, to LRP and to GradCAM.

The evaluation of XAI methods is still an open field of research [162], [163]. Therefore, an artificial evaluation data set with known ground truth was designed for quantitative evaluation. Additionally, the vibrational data from Mey et al. shown in the last chapter is used, which is missing ground truth information but can still be used for a visual, qualitative evaluation. The latter is also commonly applied [161], [164].

The proposed implementation of LIME provides more consistent results on the demo data set, as well as a more reduced number of features that are shown as significant, compared to other state of the art methods geared towards standard image explanations. Therefore, these explanations can be more beneficial when discussing model results with engineers.

The following contents are structured as follows: First, a look into related work on XAI methods will be given, particularly regarding anomaly detection in rotating machines. Afterwards, the XAI methods developed for the presented use case will be described in more detail. Finally, the method developed for evaluation will be shown.

7.1 Related Work

First, general explainable AI (XAI) approaches are described, followed by their past applications on vibration and machinery data. Lastly, evaluation approaches of XAI are discussed.

XAI algorithms aim to make models – especially Blackbox models – more understandable to humans, to avoid overfitting of a model or its focus on unimportant features and increase user trust[157], [165]–[167]. XAI methods can be categorized into model agnostic methods, backpropagation based, gradient based methods [168] and activation map-based methods. In the following, the focus will lie on the methods commonly used in vibrational analysis, i.e., LRP and GradCAM. Other methods exist in related work, for example, Deeplift [169], Deep Taylor [170], ConvNet [171], but will not be elaborated upon further in this thesis.

Model agnostic methods perturb the input data [172] and from the resulting output of the model deduce the input feature importance. Examples for this are the SHAP (SHapley Additive exPlanation) [173] and LIME (Local Interpretable Model-Agnostic Explanations) [161]. While SHAP is computationally expensive and is used for data with lesser amounts of features, it produces more accurate results [168]. LIME, on the other hand, is computationally more efficient and therefore has been used for image data [161] and for Time series [174].

LIME is used to explain the output of a model $f : \mathbb{R}^d \rightarrow \mathbb{R}$ (with d-dimensional input data) for an instance of a data set $x \in \mathbb{R}^d$ with an interpretable representation $x' \in \mathbb{R}^{d'}$. For this, multiple points z with representation z' are sampled randomly around x and thereby x' using perturbation. A second model g of a class of interpretable models G with $g : \mathbb{R}^{d'} \rightarrow \mathbb{R}$ is trained to predict the output of the original model $f(z)$ based on the perturbation information z'. Since g is interpretable, information on important perturbation features can be retrieved easily. The explanation is computed as (from [161] Equation 1)

$$\xi(x) = argmin_{g \in G} \mathcal{L}(f, g, \pi_z) + \Omega(g). \tag{7.1}$$

with the fidelity function $\mathcal{L}$ of the explanation and the complexity measure $\Omega(g)$ of g. $\mathcal{L}$ is defined as (from [161] Equation 2)

$$\mathcal{L}(f, g, \pi_z) = \sum_{z, z' \in Z} \pi_x(z)(f(z) - g(z'))^2. \tag{7.2}$$

It is to note that $\mathcal{L}$ is weighted by the proximity π_z of z to x, to focus the analysis to points that are close to the original data point. In the original paper, cosine distance is proposed as π_z for text explanations. At the same time, Ω implies that the complexity of the model g (e.g., the depth of a decision tree or the number of weights in a linear model) should be kept low. The basic principle of LIME will be modified in this chapter for transformed vibrational data.

The following methods are geared towards the internal computations of DNNs. Regarding backpropagation, a commonly used method for vibrational analysis is Layer wise Relevance Propagation (LRP) [175].

Based on the output of the final layer, the influence of its prior layer is computed using the weights of each connection of the neuron (see Figure 7.2). The influence (relevance) of each layer i's neuron is computed based on its influence on the next layer j, from the input of the network until the output neurons. The relevance of a neuron is computed as ([175] Equation 55)

$$R_i^{(l)} = \sum_j R_{i\leftarrow j}^{(l,l+1)}. \tag{7.3}$$

with the computation of the relevance R based on a stabilizing factor ϵ being for example ([175] Equation 58):

$$R_{i\leftarrow j}^{(l,l+1)} = \begin{cases} \frac{z_{ij}}{z_j+\epsilon} \cdot R_j(l+1) & z_j \geq 0 \\ \frac{z_{ij}}{z_j-\epsilon} \cdot R_j(l+1) & z_j < 0 \end{cases} \tag{7.4}$$

While this is the standard equation of LRP, there exist various kinds of LRP relevance computations which are advised to be used depending on the layer. [176], for example, instruct that LRP-0 should be used for the first layer, LRP-ϵ for the middle layers, and LRP-γ for the lower layers to improve the results. This might make generation of explanations more subjective, since the parameters are supposed to be chosen to generate the best saliency map result. In the experiments, for the vibrational data set, there was no discernible di erence in output between the variants. Therefore, only LRP-Z was used.

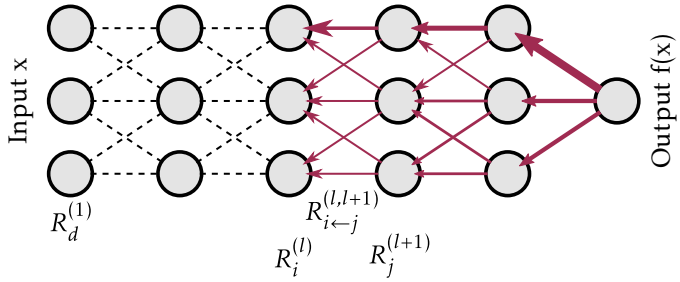


Figure 7.2: Propagation of relevance through a DNN. (Graphic after [176] Figure 10.2)

To understand GradCAM, it makes sense to first look into gradient based and activation map-based methods. Gradient based methods function similarly to backpropagation-based ones. The di erence is that instead of the network's outputs, the gradients of the output are computed and propagated backwards through the network until the input[177]. This is motivated by the fact that a high gradient is connected to an important value in the output, since changes to this point of the input would locally have a considerable influence on the networks output. Examples for gradient based methods are Integrated Gradients [178] and Smoothgrad [179].

Class Activation Maps (CAM) by Zhou et al. [180] is an algorithm where the activation maps of the layers in a CNN are weighted by their

influence on the final output. The class activation map M_c for a class c is defined by ([180] Equation 2)

$$M_c(x,y) = \sum_k w_k^c f_k(x,y) \tag{7.5}$$

with weight w of the feature map f. CAM sets requirements for the network architecture, in that it needs global-average-pooling layers for localization of the importance after the convolutional layers.

Finally, GradCAM [181] by Selvaraju et al. combines CAM and gradient based methods. It is a modification of CAM in that the feature maps are weighted using the gradient of the feature map instead of its weight.

Several pieces of related work investigate using XAI methods to get further insights on vibrational data [145]. For perturbation-based methods, Shapley values have been applied for k-NN classifier bearing faults [150]. Though, there is no related work by other authors on the application of perturbation-based method on vibrational data, yet. This chapter is the first work to adapt LIME towards the spectral use case, probably because of the computational e ort due to the high amount of input features.

It is of interest to use XAI methods for classifiers regarding physical systems' defects using vibrational data. Focusing on DNN based methods, LRP has been applied for gearbox analysis [125], and for FFT-transformed motor current analysis with CNNs [151], [158]. Kim et al. [182] use CAM for frequency transformed data for fault diagnosis. Wang et al. applied Score-CAM [183] for the rolling bearing fault diagnosis. GradCAM is another very commonly used method, for example for STFT-transformed vibrational data [153], for bearing fault analysis [160], in particular also on acoustic emission data under Hilbert transform [159], analysis of linear motion guides [184] and for power spectrum transformed bearing data [164].

Another topic of current research is the evaluation of XAI methods. For time series specifically, it is difficult to access the correctness of feature importance maps [185]. This is particularly the case if there is no ground truth in the data set on the relevant features, as is the case with the vibrational data set used [186].

In the following, the methods used in this chapter will be described in further detail, followed by the relevant experiments.

7.2 Methods

LIME was adapted towards the spectral data in the last chapter, namely frequency-RPM maps, and order-RPM maps. The algorithms used for comparison were LRP-Z implemented in the Innvestigate framework [187] and GradCAM implemented by O. Mey [188].

7.2.1 Spectral LIME

This design of the algorithm is founded upon the base implementation for LIME-for-time [174]. That method was based on data in the time domain,

using the hyperparameters of the number of segments, the length of these segments in time and the modification strategy (zero, global/segment noise, or global/segment average). In experiments using the original implementation, the segment size and number of segments changed the output of the analysis significantly.

For spectral data, e.g., the frequency and order maps from vibrational analysis, it is known that defects of the system cause changes at certain frequencies or orders in the spectra. Increases in system vibrations cause lines in the Frequency spectrum, whereas defects in elements like ball bearings or gears can lead to peaks in the order domain [135], [141]. This background knowledge can be used to analyze the complete spectral map of one class of the data. Furthermore, the perturbation of the data with zeros/noise/average causes changes to the data outside of the original data distribution, which will be shown in experiments to influence the output negatively.

Therefore, significant changes were designed to optimize the results for vibrational data in the frequency or order domain, as demonstrated in Figure 7.3. The data of one class is perturbed by randomized segments using data from the other class, before computing the prediction accuracy of the original model f on this complete data set and used as input for the simplified, linear regressor that shows the important segments of the data. This is supposed to bring the following advantages:

- The complete data of one class will be used for perturbation to improve computation time and analyze relevant di erences between classes.
- For the perturbation of one class, the data of the opposite class will be used to mitigate changes to the data distribution compared to training data.
- The algorithm is run for multiple combinations of number of segments and lengths of segments. At the end, all maps are combined using averaging. This way, the method is less sensitive on these two hyper-parameters.

An obvious disadvantage of this approach is that due to the repetition for di erent segment sizes and segment counts, this approach is computationally much less e ective than DNN-based XAI methods. Still, there are use cases that are not time-critical, such as during development of a new algorithm. In this case, slower methods like the one presented can still be beneficial.

7.2.2 Interpretable Dataset for XAI Algorithm Evaluation

To demonstrate and evaluate the presented approach, a new data set was designed which contains some of the spectral properties of real-live vibrational data, which will be described next.

With the raising popularity and capability of black box models, both casual users and domain experts require for interpretable and understandable XAI methods [168]. Due to the low average ML expertise of these

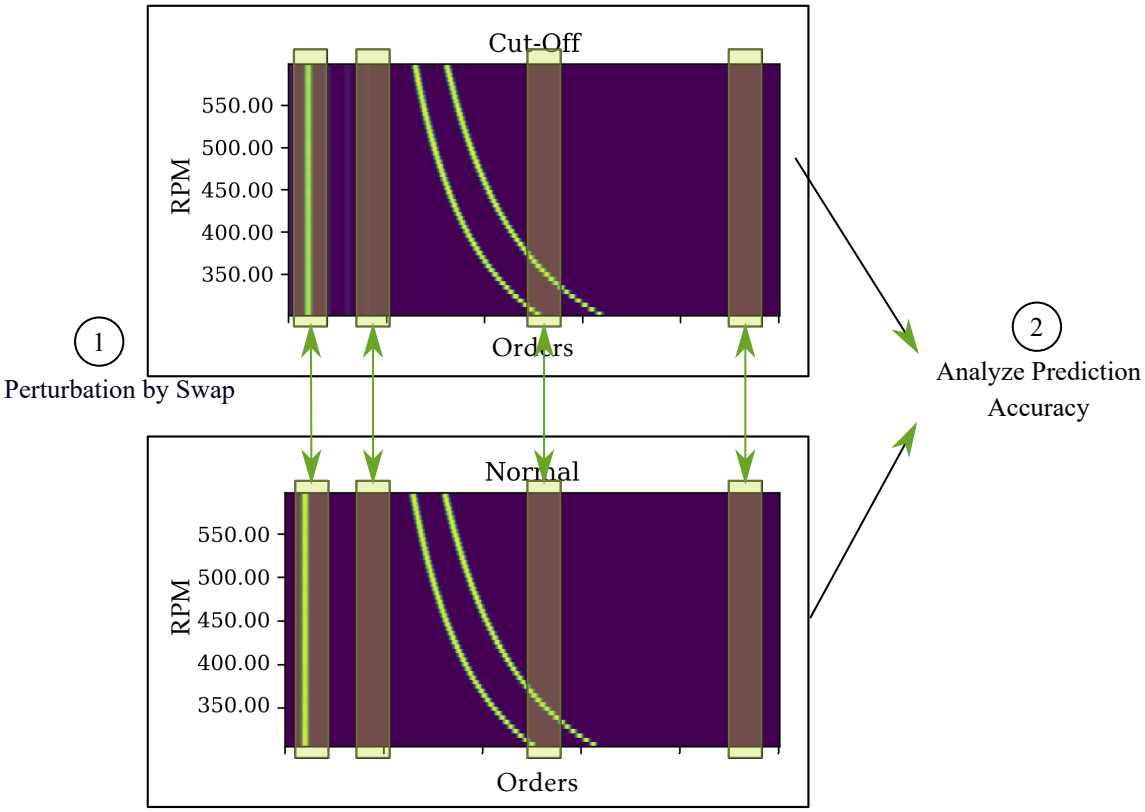


Figure 7.3: Schematic on the workings of Spectral LIME. Bands in the spectrum of one class are replaced with data at the same place of the opposite class.

groups, it is necessary to validate XAI methods thoroughly, especially since e.g., Kaur et al. [189] suggest that saliency methods can cause false trust in ML models. There is a growing body of research towards the validation of XAI methods. Kindermans et al., for example, claim that it is possible to change the explanation of a model by tweaking parameters so that the output seems more plausible to a human [190]. Still, interpreting saliency maps visually is one of the main methods of evaluation [185], which is often the case when dealing images of everyday objects.

Vibrational data make the interpretation of results more challenging because here, there often is no ground truth on the relevant frequencies or orders and evaluating vibration data requires highly specialized knowledge on data and signal processing and the underlying system itself. Therefore, as a baseline visualization of results of the proposed method of Spectral LIME, a simpler data set was designed with comparable properties to a real-life system. It is based on a sine curve with modifications that are visible in Frequency and Order maps. This way, the methods can be evaluated without having to employ additional numerical methods that, again, require further research.

The base signal of the proposed data set is a sine curve with increasing frequency over time, as shown in Figure 7.4. Two other sine signals at higher constant frequencies are added to yield the first class of the data set, the „normal“. In the second class of the data set, the stand-in for the „defective“ class, the base signal is cut o at -0.7. Figure 7.5 shows how the di erent signal components influence the spectra. In the frequency map, the base signal of the sine data set in frequency domain increases in frequency, which can be seen as the diagonal line in the order map starting at 500 RPM and frequencies under 10 Hz. The two additions at constant frequency are vertical lines in the spectrum at 60 and 80 Hz. The cuto of the data leads to additional lines in the spectrum, which follow a similar angle as the base frequency, but are lower in intensity. The order map is a normalization of the Frequency map based on the RPM of the system, which causes the visualization to be inverted. Now, the base frequency line is parallel to the y-axis of the image, and the cuto signals too. The lines that belong to the constant frequency signal now are curved. In the order transform, disturbances that correspond to the RPM of the system, like ball bearing failures or gear failures, are found as vertical lines in the order-RPM map. Damages that influence the system independent from its rotation frequency are better visible in the vertical lines in a frequency-RPM map. They cause changes in the system's resonance properties which remain at a constant position in the frequency-RPM map.

7.3 Experiments

Experiments were conducted with the model architecture from the prior chapter. The first part of the experiments was done to evaluate the perturbation strategy of Spectral LIME with the conventional ones. In the second part of the experiments, Spectral LIME is compared to LRP and GradCAM on both the sine and the vibrational data set from the prior chapter. The

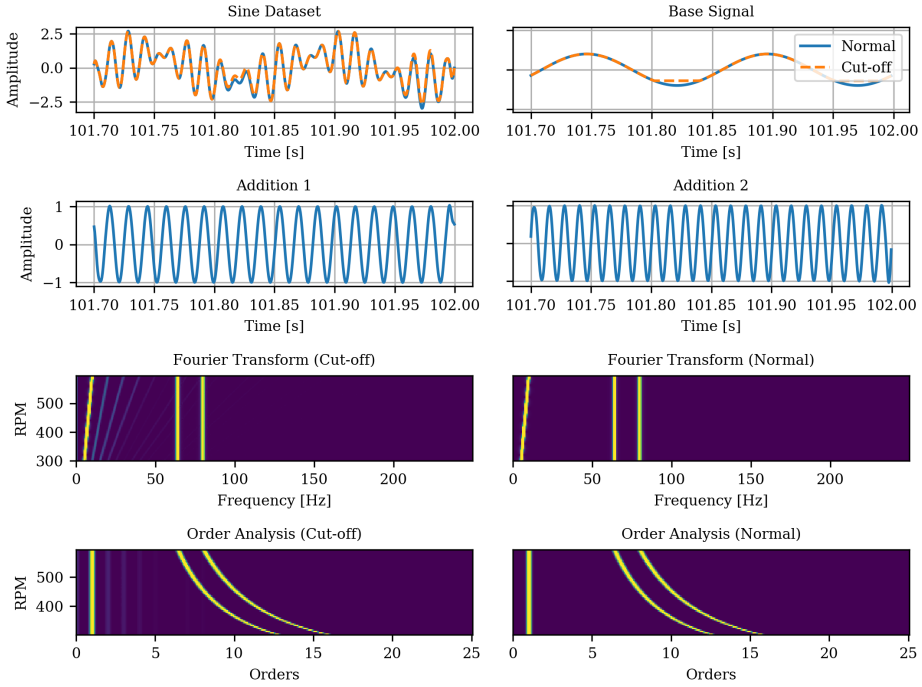


Figure 7.4: Sine data set used for evaluation in this chapter.

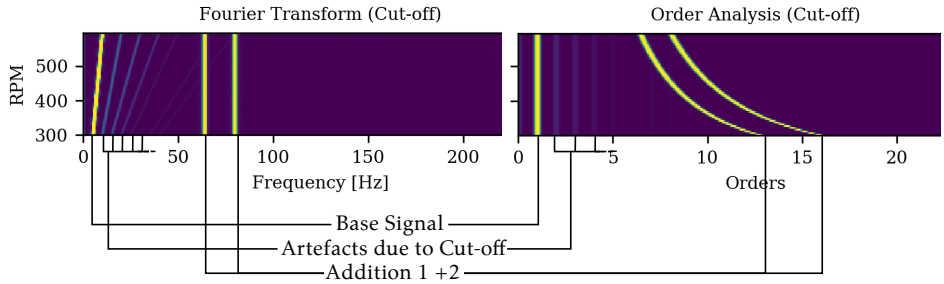


Figure 7.5: Lines in frequency- and order-RPM maps.

latter was chosen because it provided the increasing RPM necessary to create RPM maps, and because the authors undertook particular care to split the data into training, validation, and testing data sets by disassembling and reconstructing the system between recordings. A disadvantage of this data set is the missing ground truth on the relevant frequencies, though this is often the case with openly available data sets.

7.3.1 Validation of Proposed Perturbation Strategy

In this part of the experiments, LIME was applied as described in Section 7.2.1, which means that the method was run for each perturbation strategy for multiple segment widths and counts and averaged before plotting. The di erence between perturbation strategies, for example for "Mean" vs. "Total Mean", is that the Total always refers to the complete data, while the other only refers to the data in the segments that are being perturbed.

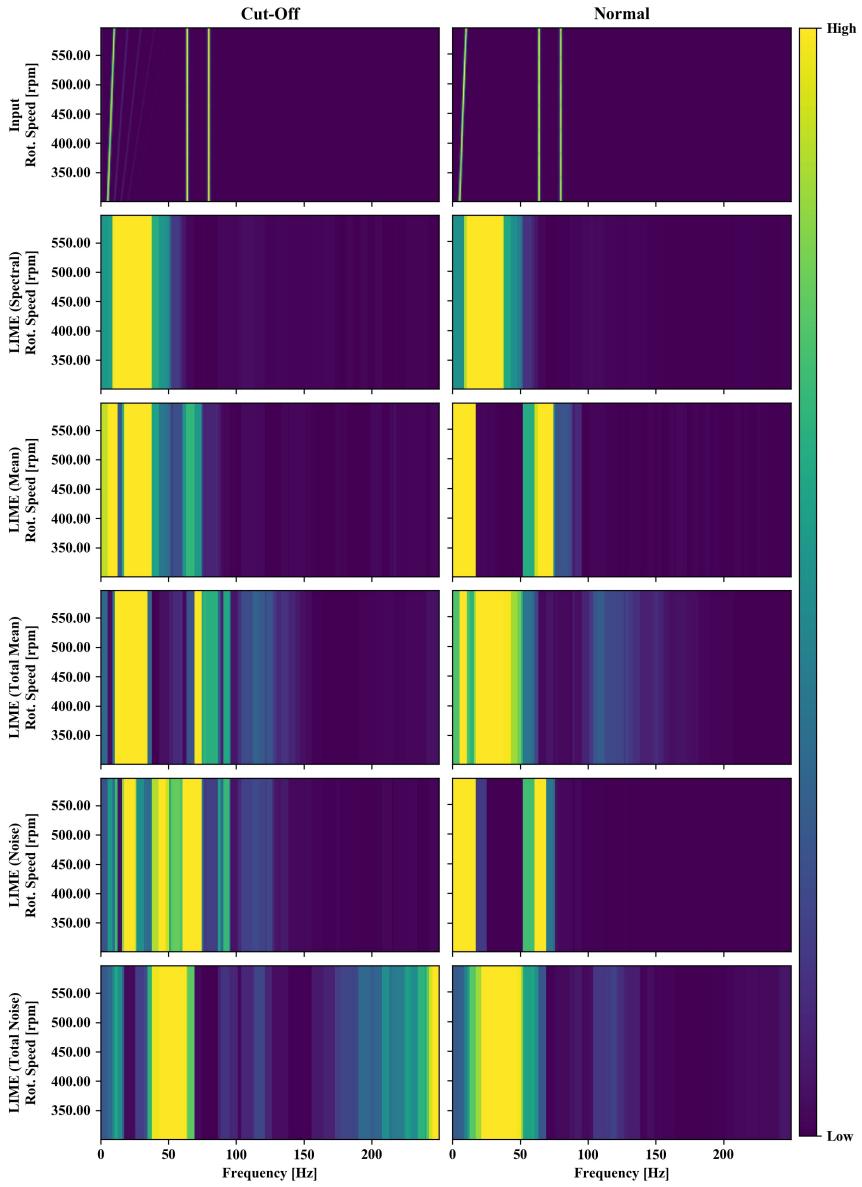
For the sine data set, results are shown in Figure 7.7, with 7.6a for the frequency-RPM map, and 7.7a for the order-RPM map. In the frequency spectrum, the Spectral LIME strategy highlights only the areas where the cuto is visible clearly, while other strategies also highlight the additions at the constant frequencies.

In the order spectrum, the situation is similar. Here, the sloped lines are highlighted in Mean and Noise, while Total Noise highlights the upper orders, where no visible di erence exists between the two classes. Another point is the consistency of results between the classes of data. While the output of Spectral LIME is similar between the classes, as is to be expected with a binary problem, the output of the other strategies is more inconsistent. This might be due to the changes to the data distribution, that these strategies apply to the data, which is mitigated when perturbing the data with data from the opposite class instead.

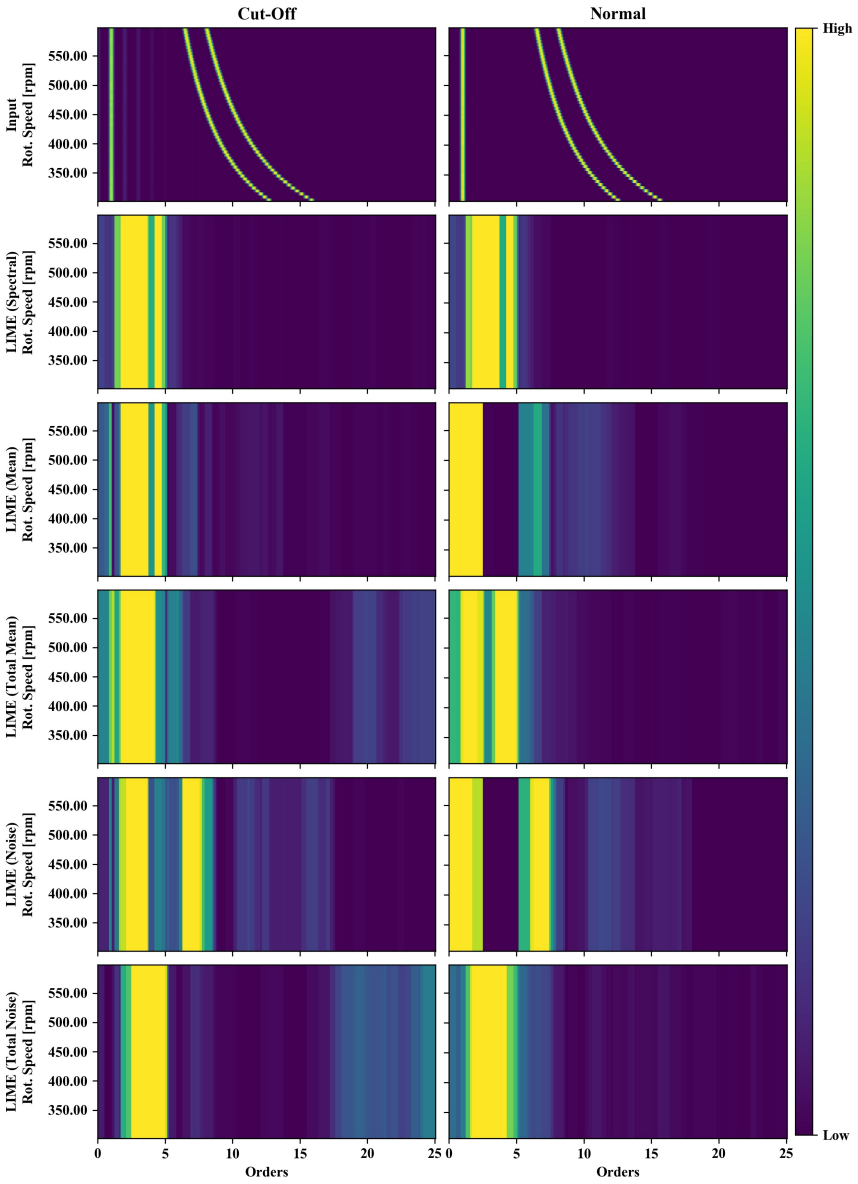
Therefore, at least for the perturbation data set, Spectral LIME produced the most consistent results of all perturbation strategies. In the next comparisons with GradCAM and LRP, only Spectral LIME will be used. Since this data set was synthetic, it is also possible to quantitatively validate the saliency maps. For this, the following terms are defined:

- **Spectral modes**: Spectral positions with top 80% of intensity.
- **Relevant pixels**: The absolute value of the subtraction of both spectra is calculated. The pixels with the top 80% values of the resulting map are referred to as **relevant pixels**
- **Irrelevant pixels** are pixels with values < 0.1 in the data from the *normal class* which are not at the same time **relevant pixels**
- **Highlighted pixels** are pixels in the heat-maps with top 80% intensity

Given this, multiple metrics were evaluated as shown in Figure 7.8. Figure 7.9a shows for the applied XAI methods the true positive rate, i.e., the number of highlighted pixels which are also relevant pixels divided by the number of relevant pixels. Global LIME and LRP-z score higher for



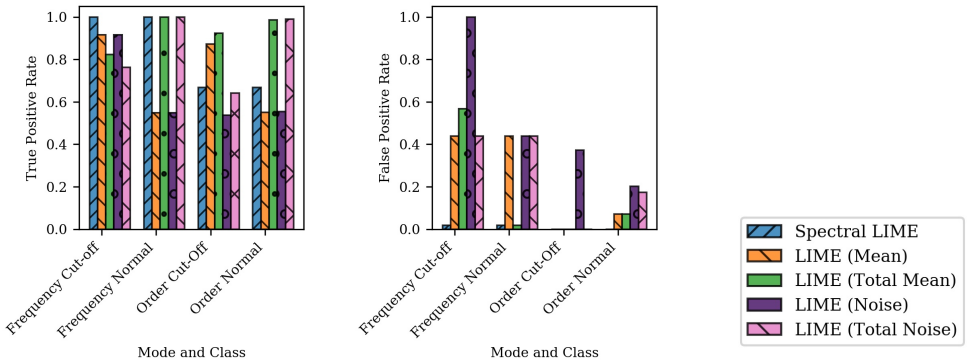
(a) Fourier transform.



(a) Order analysis.

Figure 7.7: Results of the di erent LIME perturbation strategies for the Sine data set.

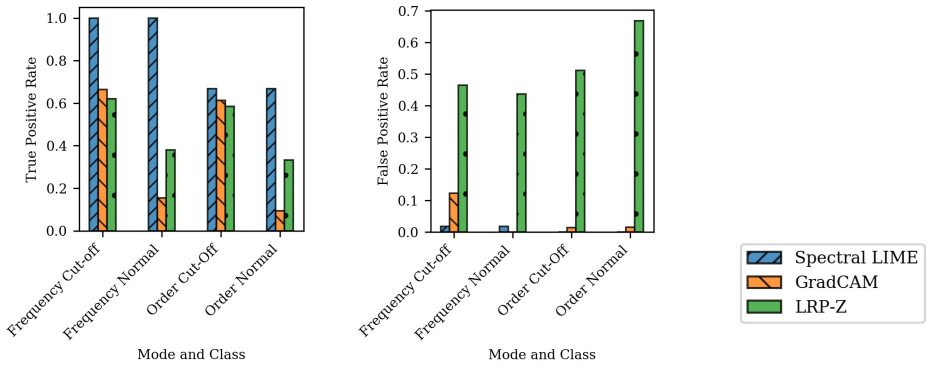
this compared to GradCAM, as could be expected from the saliency maps shown before. In contrast, Figure 7.9b shows the false positive rate, i.e., the number of highlighted pixels which are also irrelevant pixels divided by the number of irrelevant pixels. Since LRP-Z highlighted a large part of the spectral modes, it obtains high values here, while GradCAM and Global LIME have almost no false positive pixels.



(a) True positives: Ratio of pixels correctly identified as relevant.

(b) False positives: Ratio of pixels incorrectly highlighted as relevant.

Figure 7.8: Quantitative evaluation of results for sine cut-o data set for the di erent LIME variants.



(a) True positives: Ratio of pixels correctly identified as relevant.

(b) Ratio of pixels incorrectly highlighted as relevant.

Figure 7.9: Quantitative evaluation of results from Spectral LIME, GradCAM and LRP-Z for the sine cut-o data set.

7.3.2 Comparison Spectral LIME with LRP and GradCAM

Next, the results of Spectral LIME are compared with LRP and GradCAM.GradCAM and LRP were applied per line in the spectra, while Spectral LIME was used on all of the data at once. Regarding computational

performance, GradCAM and LRP was multiple orders faster than Spectral LIME.

Sine Dataset For the sine data set, results are shown in Figure 7.10, with 7.10a for the frequency-RPM map, and 7.10b for the order-RPM map.

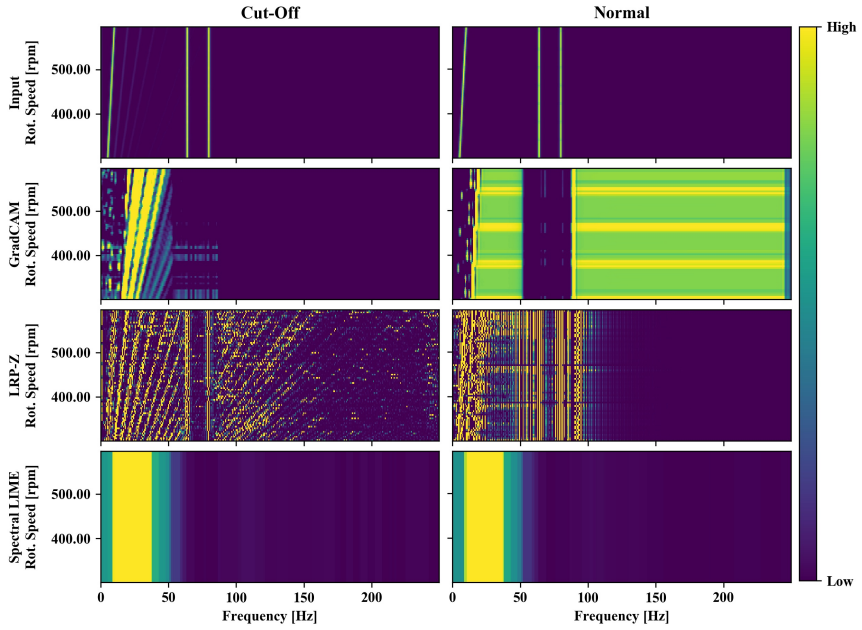
GradCAM highlights parts of the image outside of the lines of the base signals and the two additions. It is noticeable that these lines are not omitted with great accuracy, but instead with a broad margin. This is visible both in the frequency- and the order-RPM Maps. Therefore, GradCAM marks important parts of the image, but seems to not be perfect in terms of accuracy.

LRP-Z, on the other hand, seems to highlight the input data as is, with additional fine-grained noise. The vertical lines at constant frequency are highlighted as much as the rest of the data set, which shows that in this case, LRP reproduced the data set more than visualize the features that were important for discerning the classes of data.

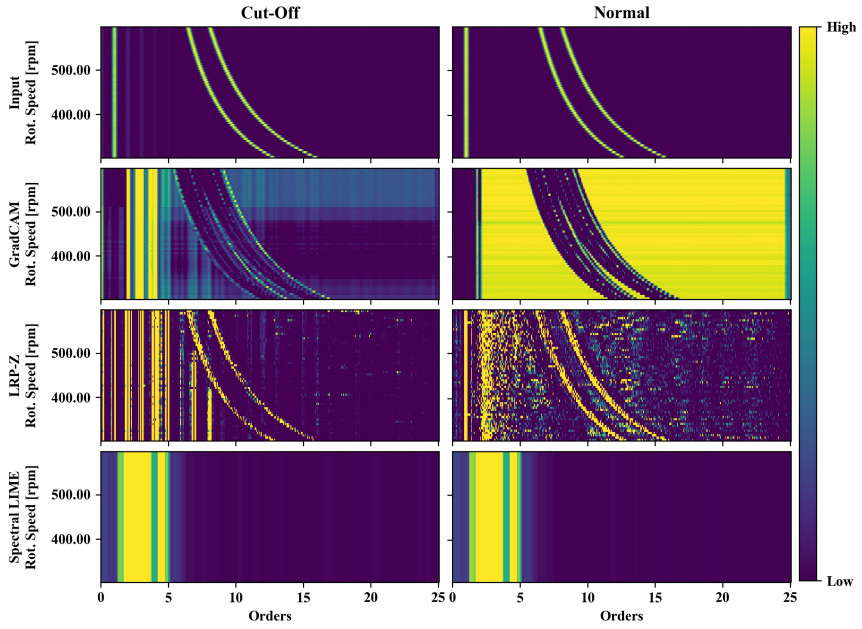
The results from Spectral LIME could already be seen in the prior section. Comparing them to the ones from LRP-Z and GradCAM shows, that for this data set, it shows the main di erence between the classes the most, i.e., that the lower frequencies and orders are important. Due to the way that Spectral LIME was designed, though, the resolution of the resulting heatmap is much lower than the other two methods, which are at pixel level.

Imbalance Dataset The results for the imbalance data set are shown in 7.11a for the frequency-RPM Map and in 7.11b for the order-RPM map. LRP-Z and Spectral LIME highlight similar parts of the data (between 0 and 500, at 800, 1200, 1500-1700). GradCAM assigns higher importance to the total imbalanced data than the balanced data. In the frequency-RPM map, the higher frequencies at the end of the spectrum seem to provide valuable information for the model, since they are highlighted by all three methods. While LIME and LRP-Z highlight similar regions, there is a significant di erence in intensity in the results of LIME, where the frequencies under 500 are assigned higher importance. This might be because these are the main system resonance frequencies that are excited by the imbalance shaking the complete system. This di erence between the classes can also be seen in the original data.

For the order-RPM maps, results appear similar. The lower orders at < 10 appear as strongest in the LIME analysis. Similar highlights can be seen in the LRP-Z results, though due to their noisiness and the limited spread in amplitude, this is not as apparent. GradCAM highlights a strip around order 12 strongly, though this is neither reproduced in LRP-z nor in LIME.

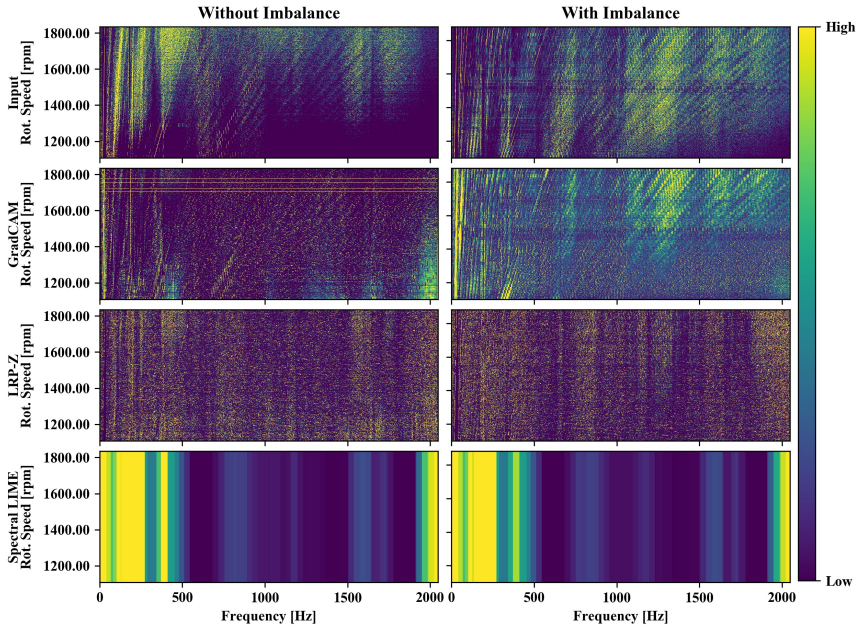


(a) Frequency-RPM Map.

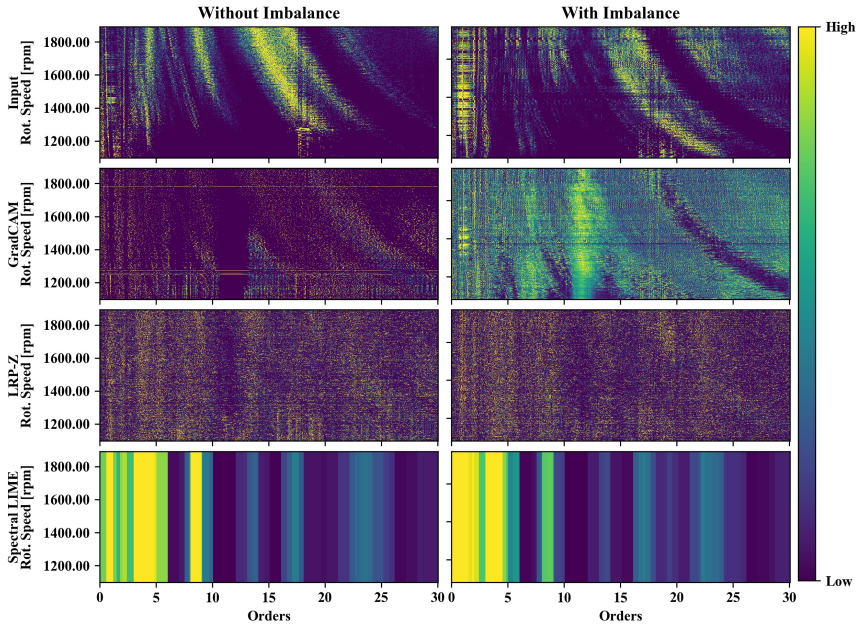


(b) Order-RPM Map.

Figure 7.10: Results of tested XAI methods for the Sine data set.



(a) Frequency-RPM Map.



(b) Order-RPM Map.

Figure 7.11: Results of tested XAI methods for the imbalance data set.

7.4 Chapter Summary

In this chapter, an adaption to LIME was presented to make spectral data, such as frequency- and order-RPM maps and their classifier models more interpretable.

Experiments were conducted with an artificial data set to validate the newly proposed perturbation strategy of Spectral LIME, where it was shown that the results were – for the given data set – more consistent than with conventional strategies. Hence, in subsequent experiments, Spectral LIME was compared to LRP-Z and GradCAM. While the results of Spectral LIME and LRP-Z showed similarities when plotted next to each other, they differed strongly from GradCAM. From the perspective of computational performance, GradCAM and LRP-Z are of advantage, because they take similar time as normal inference of a DNN. LIME, on the other hand, especially with the proposed modification, takes a long amount of time to run. It is therefore not efficient for quick, on-the-fly analysis, yet.

Unfortunately, openly available data sets for vibration analysis often do not contain all necessary information for studies like these. Often, ground truth is missing, or the sampling rate of the data is not high enough, or there is no RPM signal recorded. This poses an additional challenge to research such as this. While the artificial data set was useful for quantitative evaluation, the imbalance data set was missing ground truth information. Therefore, it was only possible to visually compare the analyses to each other and with the input data. This posed its own challenges and makes obvious that computational methods for the assessment of XAI methods [191] would be beneficial. They were knowingly not used in this chapter because they are still an open field of research [163] and would add a new layer of complexity but are an interesting topic for future research. Another open point is studying the evaluation of XAI methods with real data with the help of hardware experts. This would also be helpful in assessing the interpretability and helpfulness of an XAI method, not just the correctness.

This chapter concludes the methods investigated in this thesis. While the focus at the beginning was on the time domain of the measurement data, it was shown that frequency domain data also has its benefits in the analysis of hardware anomalies, and an explainable solution for classification was provided. This XAI method can be used to verify the classifier, as well as support experts in the analysis of the original measurements to retrieve added information about the system.

8 Conclusive Remarks and Future Work

This thesis aims to advance the state of the art of data science methods that can be of benefit for expert users for the anomaly detection for measurement data, with a focus on the domain of hydraulic test benches. Multiple pieces of original contribution to knowledge are presented, spanning over Chapters 3 to 7.

Chapter 3 presented a collection of tools for the visualization of multivariate, periodic time series data. Part of this were de-trending and o set removal, which was a novel approach for such data. Application of the proposed methods yields a first view on the data for the expert user, enabling faster interpretation of data and better visual detection of anomalies. Still, for the context of round-the-clock supervision of systems, automation of anomaly detection is necessary.

Therefore, di erent approaches for AI based anomaly detection were examined. First was a model-based approach to use recurrent neural networks for the modeling of a Digital Twin in Chapter 4. Such a model that is generated from measurements using ML, would have been the most flexible solution in case of changes in the pre-programmed system tests. It would provide benefit to other domains as well, for example in the context of system simulation. Conversely, problems were identified for the modeling of a simple hydraulic system using recurrent neural networks, which were examined in detail. Root causes and fundamental issues with the modeling of longer sequences were identified and visualized. Due to the large e ort of AI based modeling of physical systems based on measurements, this path of research was not pursued further. Therefore, and the remainder of the thesis focused on data-based methods.

Chapter 5 presents a data-oriented algorithm for unsupervised anomaly detection method for multivariate data. It assumes, that multiple systems are evaluated, and that the systems behave normally on average. Experiments were used to compare methods for data transformation, distance metrics and classifiers. The approach is designed in a way that intermediate results can be visualized for engineers and technicians, making the output interpretable and thereby simplifying root-cause-analysis. The data chosen for this approach was an openly available data set, which made it possible to examine diverse kinds of failures. Based on the data set, an incrementally failing system was also simulated and the performance of the method given a concept drift was evaluated. While the method showed high accuracy results for data in the time domain, transformation of the data to the frequency domain prior to classification yielded worse results, possibly due to the low sampling rate of the measurements or the unsuit-

ability of the method. Since spectral analysis of data is used for vibration analysis for moving systems, this led to the next point of research.

Vibrational data analysis is used, among others, for condition monitoring of rotating hardware components. Chapter 6 examines the influence of spectral analysis on the accuracy of CNN classifiers. Frequency-RPM map and order-RPM map transformed data are compared to data in the time domain. This is done in a data-based and supervised approach for the imbalance detection of a rotating system. It was shown that the preprocessing of data can improve the prediction accuracy significantly. While the classification accuracies achieved were high for the data set used, CNNs are Black Box models and lack interpretability for human users. This might impact user trust, leading to unintended consequences such as ignoring the model outputs or assigning too much confidence.

Therefore, Chapter 7, XAI approaches are discussed for this problem type. The aim of this was not only explain a model's prediction, but also enable experts to gain new insights on the data sets. For this, a new modified version of LIME [161] (Spectral LIME) was developed and compared to LRP [175] and GradCAM [162]. An easily interpretable data set with known ground truth was designed, which was used for a quantitative analysis of the results. This added another layer of insight to the qualitative analysis of the XAI methods for the vibrational data set used in the prior chapter. It was shown how the XAI algorithms highlight relevant data: Spectral LIME, for example, is built to highlight bands in the RPM maps, which corresponds to the vibrational properties of certain kinds of system components. GradCAM and LRP highlight input pixels instead. The areas highlighted di ered between the algorithms.

To summarize, this thesis investigates several aspects of the anomaly detection of physical systems under test, expanding the state of the art in the domain of data visualization, model-based approaches, and data-based approaches of di erent model complexity.

There are possible directions for future research, which are partially already discussed in each of the individual chapters. Since large parts of the experiments in this thesis were done using openly available data sets, the results are easier to reproduce than if proprietary data was used. Still, it would have been of interest to use more real-life data for evaluation. Since such data can fall under the Big Data category, further research into computationally efficient algorithms and visualizations are necessary. Especially the aspect of data reduction is important for processing substantial amounts of data in practice. For this, additional research into clustering and data reduction of multivariate time series would be beneficial. This would make data visualization, as well as anomaly detection tasks more efficient, and be used as a step used prior to the methods presented in this thesis.

Bibliography

[1] S. S. Fernández-Miranda, M. Marcos, M. E. Peralta, *et al.*, "The challenge of integrating Industry 4.0 in the degree of Mechanical Engineering", in *Manufacturing Engineering Society International Conference (MESIC'17)*, (Vigo (Pontevedra), Spain), Elsevier, Jun. 2017, pp. 1229–1236. DOI: 10.1016/j.promfg.2017.09.039.

[2] C. Bai, P. Dallasega, G. Orzes, *et al.*, "Industry 4.0 technologies assessment: A sustainability perspective", *International Journal of Production Economics*, vol. 229, no. 107776, pp. 1–15, 2020. DOI: 10.1016/j.ijpe.2020.107776.

[3] P. Kamat and R. Sugandhi, "Anomaly Detection for Predictive Maintenance in Industry 4.0- A survey", in *6th International Conference on Energy and City of the Future (EVF'19)*, (Pune City, India), vol. 170, E3S Web Conf., 2020, pp. 1–8. DOI: 10.1051/e3sconf/202017002007.

[4] Opinium, *The Human Impact of Data Literacy: A leader's guide to democratizing data, boosting productivity and empowering the workforce*, Accenture, Ed., 2020. [Online]. Available: https://web.archive.org/web/20230824093824/https://www.accenture.com/_acnmedia/PDF-115/Accenture-Human-Impact-Data-Literacy-Latest.pdf (visited on 05/13/2023).

[5] P. A. Higgs, R. Parkin, M. Jackson, *et al.*, "A Survey on Condition Monitoring Systems in Industry", in *7th Biennial Conference on Engineering Systems Design and Analysis (ASME'04)*, (Manchester, England), Online: ASME Digital Collection, 2004, pp. 163–178, ISBN: 0-7918-4175-8. DOI: 10.1115/ESDA2004-58216.

[6] H. Watter, *Hydraulik und Pneumatik: Grundlagen und Übungen – Anwendungen und Simulation*, 4th ed. Wiesbaden: Springer Vieweg Wiesbaden, 2015, ISBN: 978-3-658-07859-1. DOI: 10.1007/978-3-658-07860-7.

[7] D. Will and N. Gebhardt, *Hydraulik: Grundlagen, Komponenten, Systeme*, 6th ed. Heidelberg: Springer Vieweg Berlin, 2015, ISBN: 978-3-662-44401-6. DOI: 10.1007/978-3-662-44402-3.

[8] N. Helwig, E. Pignanelli, and A. Schütze, "Condition monitoring of a complex hydraulic system using multivariate statistics", in *IEEE International Instrumentation and Measurement Technology Conference (I2MTC'15)*, (Pisa, Italy), Dataset link: https://archive.ics.uci.edu/ml/datasets/Condition+monitoring+of+hydraulic+systems, IEEE, 2015, pp. 210–215, ISBN: 978-1-4799-6114-6. DOI: 10.1109/I2MTC.2015.7151267.

[9] W. Nelson, "Accelerated Life Testing - Step-Stress Models and Data Analyses", *IEEE Transactions on Reliability*, vol. R-29, no. 2, pp. 103–108, 1980. DOI: 10.1109/TR.1980.5220742.

[10] F. W. Spencer, "Statistical Methods in Accelerated Life Testing", *Technometrics*, vol. 33, no. 3, pp. 360–362, 1991. DOI: 10.1080/00401706.1991.10484846.

[11] H. M. Hashemian, "State-of-the-Art Predictive Maintenance Techniques", *IEEE Transactions on Instrumentation and Measurement*, vol. 60, no. 1, pp. 226–236, 2011. DOI: 10.1109/TIM.2010.2047662.

[12] K. T. Nguyen and K. Medjaher, "A new dynamic predictive maintenance framework using deep learning for failure prognostics", *Reliability Engineering & System Safety*, vol. 188, Elsevier, Ed., pp. 251–262, 2019. DOI: 10.1016/j.ress.2019.03.018.

[13] W. Zhang, D. Yang, and H. Wang, "Data-Driven Methods for Predictive Maintenance of Industrial Equipment: A Survey", *IEEE Systems Journal*, vol. 13, IEEE, Ed., pp. 2213–2227, 2019. DOI: 10.1109/JSYST.2019.2905565.

[14] O. Mey and D. Neufeld, "Explainable AI Algorithms for Vibration Data-Based Fault Detection: Use Case-Adadpted Methods and Critical Evaluation", *Sensors*, vol. 22, MDPI, Ed., pp. 1–22, 2022, Source Code link: https://github.com/o-mey/xai-vibration-fault-detection. DOI: 10.3390/s22239037.

[15] D. Neufeld and U. Schmid, "Anomaly Detection for Hydraulic Systems under Test", in *IEEE 26th International Conference on Emerging Technologies and Factory Automation (ETFA'21)*, (Vasteras, Sweden), IEEE, 2021, pp. 1–8, ISBN: 978-1-7281-2989-1. DOI: 10.1109/ETFA45728.2021.9613265.

[16] E. Y. Gorodov and V. V. Gubarev, "Analytical Review of Data Visualization Methods in Application to Big Data", *Journal of Electrical and Computer Engineering*, vol. 2013, Hindawi, Ed., pp. 1–7, DOI: 10.1155/2013/969458.

[17] K. G. Mehrotra, C. K. Mohan, and H. Huang, "Model-based anomaly detection approaches", in *Anomaly Detection Principles and Algorithms*, ser. Terrorism, Security, and Computation, K. G. Mehrotra, C. K. Mohan, and H. Huang, Eds., Cham: Springer International Publishing, 2017, pp. 57–94, ISBN: 978-3-319-67524-4. DOI: \url{10.1007/978-3-319-67526-8_5}.

[18] D. Neufeld, "Visualization Methods for Periodic Time Series Data", in *Lernen. Wissen. Daten. Analysen. (LWDA'21)*, (Munich, Germany), Online: CEUR Workshop Proceedings, Sep. 2021, pp. 1–10. [Online]. Available: https://ceur-ws.org/Vol-2993/paper-16.pdf.

[19] Standards Coordinating Committee of the IEEE Computer Society, *IEEE standard computer dictionary: A compilation of IEEE standard computer glossaries, 610*. New York, NY, USA: IEEE, 1990, vol. 610, ISBN: 1-55937-079-3. DOI: 10.1109/IEEESTD.1991.106963.

[20] B. Bertsche, *Reliability in automotive and mechanical engineering: Determination of component and system reliability*. Berlin and Heidelberg: Springer, 2008, ISBN: 978-3-540-34282-3. DOI: 10.1007/978-3-540-34282-3.

[21] D. H. Collins, J. K. Freels, A. V. Huzurbazar, *et al.*, "Accelerated Test Methods for Reliability Prediction", *Journal of Quality Technology*, vol. 45, pp. 244–259, 2013. DOI: 10.1080/00224065.2013.11917936.

[22] M. Deistler and W. Scherrer, *Modelle der Zeitreihenanalyse*. Cham: Springer International Publishing, 2018, ISBN: 978-3-319-68663-9. DOI: \url{10.1007/978-3-319-68664-6}.

[23] A. A. Cook, G. Misirli, and Z. Fan, "Anomaly Detection for IoT Time-Series Data: A Survey", *IEEE Internet of Things Journal*, vol. 7, IEEE, Ed., pp. 6481–6494, 2019. DOI: 10.1109/JIOT.2019.2958185.

[24] M. Fahim and A. Sillitti, "Anomaly Detection, Analysis and Prediction Techniques in IoT Environment: A Systematic Literature Review", *IEEE Access*, vol. 7, IEEE, Ed., pp. 81 664–81 681, 2019. DOI: 10.1109/ACCESS.2019.2921912.

[25] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey", *ACM Computing Surveys*, vol. 41, pp. 1–58, 2009. DOI: 10.1145/1541880.1541882.

[26] N. Saravanan, V. K. Siddabattuni, and K. I. Ramachandran, "Fault diagnosis of spur bevel gear box using artificial neural network (ANN), and proximal support vector machine (PSVM)", *Applied Soft Computing*, no. 10, pp. 344–360, 2010. DOI: 10.1016/j.asoc.2009.08.006.

[27] H. Dhiman, D. Deb, S. M. Muyeen, *et al.*, "Wind Turbine Gearbox Anomaly Detection Based on Adaptive Threshold and Twin Support Vector Machines", *IEEE Transactions on Energy Conversion*, vol. 36, no. 4, pp. 3462–3469, 2021. DOI: 10.1109/TEC.2021.3075897.

[28] F. P. García Márquez, A. M. Tobias, J. M. Pinar Pérez, *et al.*, "Condition monitoring of wind turbines: Techniques and methods", *Renewable Energy*, vol. 46, pp. 169–178, 2012. DOI: 10.1016/j.renene.2012.03.003.

[29] G. Jiang, P. Xie, H. He, *et al.*, "Wind Turbine Fault Detection Using a Denoising Autoencoder With Temporal Information", *IEEE/ASME Transactions on Mechatronics*, vol. 23, no. 1, pp. 89–100, 2018, ISSN: 1083-4435. DOI: 10.1109/TMECH.2017.2759301.

[30] J. Lee, F. Wu, W. Zhao, *et al.*, "Prognostics and health management design for rotary machinery systems—Reviews, methodology and applications", *Mechanical Systems and Signal Processing*, vol. 42, no. 1-2, pp. 314–334, 2014, ISSN: 08883270. DOI: 10.1016/j.ymssp.2013.06.004.

[31] H. Yan, J. Sun, and H. Zuo, "Anomaly detection based on multivariate data for the aircraft hydraulic system", *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, vol. 235, no. 5, pp. 593–605, 2021, ISSN: 0959-6518. DOI: 10.1177/0959651820954577.

[32] N. Helwig, E. Pignanelli, and A. Schütze, "D8.1 - Detecting and Compensating Sensor Faults in a Hydraulic Condition Monitoring System", in *AMA Conferences (2015)*, (Nuremberg, Germany), AMA Service GmbH, 2015, pp. 641–646, ISBN: 978-3-9813484-8-4. DOI: 10.5162/sensor2015/D8.1.

[33] Y. Jia, M. Xu, and R. Wang, "Symbolic Important Point Perceptually and Hidden Markov Model Based Hydraulic Pump Fault Diagnosis Method", *Sensors*, vol. 18, no. 12, pp. 1–20, 2018. DOI: 10.3390/s18124460.

[34] J. Hochenbaum, O. S. Vallis, and A. Kejariwal, "Automatic anomaly detection in the cloud via statistical learning", *CoRR*, pp. 1–13, 2017, Dataset / Source Code link: https://github.com/twitter/AnomalyDetection/. arXiv: 1704.07706.

[35] H. Xu, Y. Feng, J. Chen, *et al.*, "Unsupervised Anomaly Detection via Variational Auto-Encoder for Seasonal KPIs in Web Applications", in *World Wide Web Conference (WWW'18)*, (Lyon, France), New York, New York, USA: ACM Press, 2018, pp. 187–196, ISBN: 9781450356398. DOI: 10.1145/3178876.3185996.

[36] R. Assaf, I. Giurgiu, J. Pfe erle, *et al.*, "An Anomaly Detection and Explainability Framework using Convolutional Autoencoders for Data Storage Systems", in *29th International Joint Conference on Artificial Intelligence Demos (IJCAI-PRICAI'20)*, (Yokohama, Japan), International Joint Conferences on Artificial Intelligence Organization, 2020, pp. 5228–5230, ISBN: 978-0-9992411-6-5. DOI: 10.24963/ijcai.2020/752.

[37] S. Nandi, H. A. Toliyat, and X. Li, "Condition Monitoring and Fault Diagnosis of Electrical Motors—A Review", *IEEE transactions on Energy Conversion*, vol. 20, no. 4, pp. 719–729, 2005, ISSN: 0885-8969. DOI: 10.1109/tec.2005.847955.

[38] J. Lacaille, V. Gerez, and R. Zouari, "An Adaptive Anomaly Detector used in Turbofan Test Cells", in *Annual Conference of the PHM Society*, (Portland, Oregon), PHM Society, 2010, pp. 1–9. DOI: 10.36001/phmconf.2010.v2i1.1865.

[39] W. Jiang, S. K. Spurgeon, J. A. Twiddle, *et al.*, "A wavelet cluster-based band-pass filtering and envelope demodulation approach with application to fault diagnosis in a dry vacuum pump", *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, vol. 221, no. 11, pp. 1279–1286, 2007. DOI: 10.1243/09544062JMES544.

[40] J. W. Tukey, *Exploratory data analysis*, [Nachdr.], ser. Addison-Wesley series in behavioral science. Reading, Mass. u.a.: Addison-Wesley, 1998, ISBN: 978-0201076165. [Online]. Available: https://permalink.obvsg.at/AC02560792.

[41] L. A. Kurgan and P. Musilek, "A survey of Knowledge Discovery and Data Mining process models", *The Knowledge Engineering Review*, vol. 21, no. 1, pp. 1–24, 2006, ISSN: 0269-8889. DOI: 10.1017/S0269888906000737.

[42] U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, "From Data Mining to Knowledge Discovery in Databases", *AI Magazine*, vol. 17, pp. 37–54, 1996. DOI: 10.1609/aimag.v17i3.1230.

[43] A. Carreño, I. Inza, and J. A. Lozano, "Analyzing rare event, anomaly, novelty and outlier detection terms under the supervised classification framework", *Artificial Intelligence Review*, vol. 53, no. 5, pp. 3575–3594, 2020, ISSN: 0269-2821. DOI: 10.1007/s10462-019-09771-y.

[44] M. Goldstein and S. Uchida, "A Comparative Evaluation of Unsupervised Anomaly Detection Algorithms for Multivariate Data", *PloS one*, no. 11, p. 31, 2016, ISSN: 1932-6203. DOI: 10.1371/journal.pone.0152173.

[45] M. Landry, F. Leonard, C. Landry, *et al.*, "An Improved Vibration Analysis Algorithm as a Diagnostic Tool for Detecting Mechanical Anomalies on Power Circuit Breakers", *IEEE Transactions on Power Delivery*, vol. 23, no. 4, pp. 1986–1994, 2008, ISSN: 0885-8977. DOI: 10.1109/TPWRD.2008.2002846.

[46] M. Munir, S. A. Siddiqui, A. Dengel, *et al.*, "DeepAnT: A Deep Learning Approach for Unsupervised Anomaly Detection in Time Series", *IEEE Access*, vol. 7, pp. 1991–2005, 2019. DOI: 10.1109/ACCESS.2018.2886457.

[47] J. Sipple, "Interpretable, Multidimensional, Multimodal Anomaly Detection with Negative Sampling for Detection of Device Failure", in *37th International Conference on Machine Learning*, (Virtual), vol. 119, MLResearchPress, 2020, pp. 9016–9025. [Online]. Available: https://proceedings.mlr.press/v119/sipple20a.html.

[48] M. W. Grieves, "Virtually intelligent product systems: Digital and physical twins", in *Complex Systems Engineering: Theory and Practice*. Reston, VA: American Institute of Aeronautics and Astronautics, Inc, 2019, vol. 411, pp. 175–200, ISBN: 978-1-62410-564-7. DOI: 10.2514/5.9781624105654.0175.0200.

[49] M. Kordestani, M. Saif, M. E. Orchard, *et al.*, "Failure Prognosis and Applications—A Survey of Recent Literature", *IEEE Transactions on Reliability*, vol. 70, no. 2, pp. 728–748, 2021, ISSN: 0018-9529. DOI: 10.1109/TR.2019.2930195.

[50] M. Yadav, P. Malhotra, L. Vig, *et al.*, "ODE - augmented training improves anomaly detection in sensor data from machines", *CoRR*, vol. abs/1605.01534, 2016. arXiv: 1605.01534.

[51] N. Laptev, S. Amizadeh, and I. Flint, "Generic and Scalable Framework for Automated Time-series Anomaly Detection", in *21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'15)*, (Sydney, Australia), New York, NY, USA: ACM Digital Library, Aug. 2015, pp. 1939–1947. DOI: 10.1145/2783258.2788611.

[52] F. Immovilli, M. Cocconcelli, A. Bellini, *et al.*, "Detection of Generalized-Roughness Bearing Fault by Spectral-Kurtosis Energy of Vibration or Current Signals", *IEEE Transactions on Industrial Electronics*, vol. 56, no. 11, pp. 4710–4717, 2009, ISSN: 0278-0046. DOI: 10.1109/TIE.2009.2025288.

[53] M. M. Breunig, H.-P. Kriegel, R. T. Ng, *et al.*, "LOF: identifying density-based local outliers", in *ACM International Conference on Management of Data and Symposium on Principles of Database Systems (SIGMOD'00)*, (New York, New York), New York, New York, USA: ACM Press, 2000, pp. 93–104, ISBN: 1581132174. DOI: 10.1145/342009.335388.

[54] K. Beyer, J. Goldstein, R. Ramakrishnan, *et al.*, "When Is "Nearest Neighbor" Meaningful?", in *Proceedings on International Conference on Database Theory (ICDT'1999)*, ser. Lecture Notes in Computer Science, vol. 1540, Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 217–235, ISBN: 978-3-540-65452-0. DOI: 10.1007/3-540-49257-7_15.

[55] A. Kejariwal, *Introducing practical and robust anomaly detection in a time series*, 2015. [Online]. Available: https://web.archive.org/web/20210506134624/https://blog.twitter.com/%5C%5Cengineering/en_us/a/2015/introducing-practical-and-robust-anomaly-detection-in-a-time-series.html (visited on 05/06/2021).

[56] W. Aigner, S. Miksch, H. Schumann, *et al.*, *Visualization of Time-Oriented Data*. London, UK: Springer London, 2011, ISBN: 978-0-85729-078-6. DOI: 10.1007/978-0-85729-079-3.

[57] P. Virtanen, R. Gommers, T. E. Oliphant, *et al.*, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python", *Nature Methods*, vol. 17, pp. 261–272, 2020. DOI: 10.1038/s41592-019-0686-2.

[58] G. Chakraborty, T. Kamiyama, H. Takahashi, *et al.*, "An Efficient Anomaly Detection in Quasi-Periodic Time Series Data—A Case Study with ECG", in *International Conference on Time Series (ITISE'17)*, (Granada, Spain), Cham, Germany: Springer International Publishing, 2018, pp. 147–157, ISBN: 978-3-319-96943-5. DOI: 10.1007/978-3-319-96944-2_10.

[59] T. Schneider, N. Helwig, and A. Schütze, "Automatic feature extraction and selection for classification of cyclical time series data", *tm - Technisches Messen*, vol. 84, no. 3, pp. 198–206, 2017, ISSN: 0171-8096. DOI: 10.1515/teme-2016-0072.

[60] R. Wu and E. J. Keogh, "Current time series anomaly detection benchmarks are flawed and are creating the illusion of progress", *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 03, pp. 2421–2429, 2023, ISSN: 1558-2191. DOI: 10.1109/TKDE.2021.3112126.

[61] Y. Fang, H. Xu, and J. Jiang, "A Survey of Time Series Data Visualization Research", in *3rd International Conference on Energy Material, Chemical Engineering and Mining Engineering (EMCEME'19)*, (Qingdao, China), ser. IOP Conference Series: Materials Science and Engineering, vol. 782, IOP Publishing Ltd, 2020, p. 10. DOI: 10.1088/1757-899X/782/2/022013.

[62] J. J. van Wijk and E. R. van Selow, "Cluster and calendar based visualization of time series data", in *1999 IEEE Symposium on Information Visualization (InfoVis'99)*, (San Francisco, CA, USA), IEEE Comput. Soc, 1999, pp. 4–9, ISBN: 0-7695-0431-0. DOI: 10.1109/INFVIS.1999.801851.

[63] C. Macas and P. Machado, "Radial Calendar of Consumption", in *22nd International Conference Information Visualisation (IV'18)*, (Fisciano, Italy), IEEE, 2018, pp. 96–102, ISBN: 978-1-5386-7202-0. DOI: 10.1109/iV.2018.00027.

[64] P. Buono and F. Balducci, "MonitorApp: a web tool to analyze and visualize pollution data detected by an electronic nose", *Multimedia Tools and Applications*, vol. 78, no. 23, pp. 33 023–33 040, 2019, ISSN: 1380-7501. DOI: 10.1007/s11042-019-7676-3.

[65] K. Matkovic, D. Gracanin, M. Jelovic, *et al.*, "Interactive visual analysis of large simulation ensembles", in *2015 Winter Simulation Conference (WSC)*, (Huntington Beach, CA, USA), IEEE, 2016, pp. 517–528, ISBN: 978-1-4673-9743-8. DOI: 10.1109/WSC.2015.7408192.

[66] E. Gjika, L. Basha, A. Ferrja, *et al.*, "Analyzing Seasonality in Hydropower Plants Energy Production and External Variables", in *The 7th International conference on Time Series and Forecasting (ITISE'21)*, (Gran Canaria, Spain), Basel Switzerland: MDPI, Jul. 2021, p. 15. DOI: 10.3390/engproc2021005015.

[67] J. Lin, E. Keogh, S. Lonardi, *et al.*, "VizTree A Tool for Visually Mining and Monitoring Massive Time Series Databases", in *Proceedings 30th Annual International Conference on Very Large Data Bases (VLDB'04)*, Elsevier, Sep. 2004, pp. 1269–1272, ISBN: 978-0-12-088469-8. DOI: 10.1016/B978-012088469-8/50124-8.

[68] R. J. Hyndman and H. L. Shang, "Rainbow Plots, Bagplots, and Boxplots for Functional Data", *Journal of Computational and Graphical Statistics*, vol. 19, no. 1, pp. 29–45, 2010, ISSN: 1061-8600. DOI: 10.1198/jcgs.2009.08158.

[69] J. S. Yi, Y. A. Kang, J. Stasko, *et al.*, "Toward a deeper understanding of the role of interaction in information visualization", *IEEE transactions on visualization and computer graphics*, vol. 13, no. 6, pp. 1224–1231, 2007, ISSN: 1077-2626. DOI: 10.1109/TVCG.2007.70515.

[70] M. Wertheimer, "Untersuchungen zur Lehre von der Gestalt. II", *Psychologische Forschung*, vol. 4, no. 1, pp. 301–350, 1923, ISSN: 0340-0727. DOI: 10.1007/BF00410640.

[71] J. D. Hunter, "Matplotlib: A 2D graphics environment", *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007. DOI: 10.1109/MCSE.2007.55.

[72] A. Lee, *Mplcursors*, 2021. [Online]. Available: https://mplcursors.readthedocs.io/en/stable/ (visited on 05/06/2021).

[73] Anthony Atkielski, *File:SinusRhythmLabels.svg*, 2007. [Online]. Available: https://commons.wikimedia.org/wiki/File:SinusRhythmLabels.svg (visited on 03/30/2022).

[74] A. Birolini, *Reliability Engineering – Theory and Practice*. Dordrecht: Springer, 2013, p. 502, ISBN: 978-3-662-03792-8. DOI: 10.1007/978-3-662-03792-8.

[75] E. Negri, L. Fumagalli, and M. Macchi, "A Review of the Roles of Digital Twin in CPS-based Production Systems", *Procedia Manufacturing*, vol. 11, pp. 939–948, 2017, ISSN: 23519789. DOI: 10.1016/j.promfg.2017.07.198.

[76] J. Gama, I. Žliobaitė, A. Bifet, *et al.*, "A survey on concept drift adaptation", *ACM Computing Surveys*, vol. 46, no. 4, pp. 1–37, 2014, ISSN: 0360-0300. DOI: 10.1145/2523813.

[77] D. Chicco, "Ten quick tips for machine learning in computational biology", *BioData mining*, vol. 10, p. 35, 2017, ISSN: 1756-0381. DOI: 10.1186/s13040-017-0155-3.

[78] Tony R. Kuphaldt, *Lessons In Industrial Instrumentation: Chapter 15 - Basic Principles of Instrument Calibration and Ranging: Calibration Errors and Testing*, Creative Commons Attribution 4.0 International Public License, Ed., 2018. [Online]. Available: https://control.com/textbook/introduction-to-industrial-instrumentation/ (visited on 06/23/2022).

[79] M. I. Jordan, "Attractor Dynamics and Parallelism in a Connectionist Sequential Machine", in *Artificial Neural Networks: Concept Learning*, IEEE Press, 1990, pp. 112–127, ISBN: 0818620153.

[80] J. L. Elman, "Finding Structure in Time", *Cognitive Science*, vol. 14, no. 2, pp. 179–211, 1990, ISSN: 03640213. DOI: 10.1207/s15516709cog1402_1.

[81] S. Hochreiter and J. Schmidhuber, "Long short-term memory", *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997, ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735.

[82] K. Cho, B. van Merriënboer, C. Gulcehre, *et al.*, "Learning phrase representations using RNN encoder–decoder for statistical machine translation", in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP'14)*, (Doha, Qatar), Association for Computational Linguistics, Oct. 2014, pp. 1724–1734. DOI: 10.3115/v1/D14-1179.

[83] F.-J. Chang, P.-A. Chen, Y.-R. Lu, *et al.*, "Real-time multi-step-ahead water level forecasting by recurrent neural networks for urban flood control", *Journal of Hydrology*, vol. 517, pp. 836–846, 2014, ISSN: 00221694. DOI: 10.1016/J.JHYDROL.2014.06.013.

[84] M. Cheroutre-Vialette and A. Lebert, "Application of recurrent neural network to predict bacterial growth in dynamic conditions", *International Journal of Food Microbiology*, vol. 73, no. 2-3, pp. 107–118, 2002, ISSN: 01681605. DOI: 10.1016/S0168-1605(01)00642-0.

[85] J. Chung, Ç. Gülçehre, K. Cho, *et al.*, "Empirical evaluation of gated recurrent neural networks on sequence modeling", *CoRR*, vol. abs/1412.3555, p. 9, 2014. arXiv: 1412.3555.

[86] J. W. Forrester, "Dynamic models of economic systems and industrial organizations", *System Dynamics Review*, vol. 19, no. 4, pp. 329–345, 2003, ISSN: 08837066. DOI: 10.1002/sdr.284.

[87] N. Ampilova, V. Sergeev, and I. Soloviev, "On application of dynamical system methods in biomedical engineering", *Vibroengineering PROCEDIA*, vol. 26, pp. 52–56, 2019, ISSN: 2345-0533. DOI: 10.21595/vp.2019.20972.

[88] O. Föllinger, U. Konigorski, B. Lohmann, *et al.*, *Regelungstechnik: Einführung in die Methoden und ihre Anwendung*, 12., überarbeitete Auflage, ser. Lehrbuch Studium. Berlin and O enbach: VDE Verlag GmbH, 2016, ISBN: 978-3-8007-4201-1.

[89] S. L. Brunton, J. L. Proctor, and J. N. Kutz, "Discovering governing equations from data by sparse identification of nonlinear dynamical systems", *Proceedings of the National Academy of Sciences of the United States of America*, vol. 113, no. 15, pp. 3932–3937, 2016. DOI: 10.1073/pnas.1517384113.

[90] R. Isermann and M. Münchhof, *Identification of Dynamic Systems*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, ISBN: 978-3-540-78878-2. DOI: 10.1007/978-3-540-78879-9.

[91] T. Q. Chen, Y. Rubanova, J. Bettencourt, *et al.*, "Neural ordinary di erential equations", in *32nd International Conference on Neural Information Processing Systems (NIPS'18)*, (Montréal Canada), ser. NIPS'18, vol. abs/1806.07366, Red Hook, NY, USA: Curran Associates Inc., 2018, pp. 6572–6583. arXiv: 1806.07366.

[92] J. Wilczek, A. Wright, V. Välimäki, *et al.*, "Virtual analog modeling of distortion circuits using neural ordinary di erential equations", in *25th International Conference on Digital Audio E ects (DAFx20in22)*, (Vienna, Austria), 2022, pp. 9–16. arXiv: 2205.01897 [eess.AS].

[93] J. Brucker, W. G. Bessler, and R. Gasper, "Grey-box modelling of lithium-ion batteries using neural ordinary di erential equations", *Energy Informatics*, vol. 4, no. S3, p. 13, 2021. DOI: 10.1186/s42162-021-00170-8.

[94] R. Valle, F. A. Reda, M. Shoeybi, *et al.*, "Neural ODEs for Image Segmentation with Level Sets", *CoRR*, vol. abs/1912.11683, 2019. arXiv: 1912.11683.

[95] K. Gupta and M. Chandraker, "Neural mesh flow: 3d manifold mesh generationvia di eomorphic flows", *CoRR*, vol. abs/2007.10973, 2020. arXiv: 2007.10973.

[96] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning Internal Representations by Error Propagation", *Readings in Cognitive Science*, pp. 399–421, 1988. DOI: 10.1016/B978-1-4832-1446-7.50035-2.

[97] M. I. Jordan, "Serial order: A parallel distributed processing approach. technical report, june 1985-march 1986", May 1986. [Online]. Available: https://www.osti.gov/biblio/6910294.

[98] C.-W. Chang, M. Ushio, and C.-h. Hsieh, "Empirical dynamic modeling for beginners", *Ecological Research*, vol. 32, no. 6, pp. 785–796, 2017, ISSN: 0912-3814. DOI: 10.1007/s11284-017-1469-9.

[99] M. Zounemat-Kermani, E. Matta, A. Cominola, *et al.*, "Neurocomputing in surface water hydrology and hydraulics: A review of two decades retrospective, current status and future prospects", *Journal of Hydrology*, vol. 588, p. 17, 2020, ISSN: 00221694. DOI: 10.1016/j.jhydrol.2020.125085.

[100] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling", in *Interspeech 2014*, (Singapore), ISCA: ISCA, Sep. 2014, pp. 338–342. DOI: 10.21437/Interspeech.2014-80.

[101] P. Rodriguez, J. Wiles, and J. L. Elman, "A Recurrent Neural Network that Learns to Count", *Connection Science*, vol. 11, no. 1, pp. 5–40, 1999, ISSN: 0954-0091. DOI: 10.1080/095400999116340.

[102] M. Suzgun, Y. Belinkov, S. Shieber, *et al.*, "LSTM Networks Can Perform Dynamic Counting", pp. 44–54, 2019. DOI: 10.18653/v1/W19-3905.

[103] F. A. Gers and J. Schmidhuber, "Recurrent nets that time and count", in *International Joint Conference on Neural Networks. (IJCNN 2000). Neural Computing: New Challenges and Perspectives for the New Millennium*, (Como, Italy), IEEE, 2000, 189–194 vol.3, ISBN: 0-7695-0619-4. DOI: 10.1109/IJCNN.2000.861302.

[104] J. Wiles and J. L. Elman, "Learning to count without a counter: A case study of dynamics and activation landscapes in recurrent networks", 1995. [Online]. Available: https://www.researchgate.net/publication/2252761_Learning_to_count_without_a_counter_A_case_study_of_dynamics_and_activation_landscapes_in_recurrent_networks (visited on 06/12/2022).

[105] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult", *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994, ISSN: 1045-9227. DOI: 10.1109/72.279181.

[106] Roger Grosse, *Lecture 15: Exploding and Vanishing Gradients*, Lecture Notes, Toronto, 2017. [Online]. Available: https://web.archive.org/web/20230612175058/https://www.cs.toronto.edu/~rgrosse/courses/csc321_2017/readings/L15%20Exploding%20and%20Vanishing%20Gradients.pdf (visited on 06/12/2022).

[107] Martín Abadi, Ashish Agarwal, Paul Barham, *et al.*, *TensorFlow: Large-scale machine learning on heterogeneous systems*, Software available from tensorflow.org, 2015. [Online]. Available: https://www.tensorflow.org/.

[108] D. P. K. and Jimmy Ba, "Adam: A method for stochastic optimization", in *3rd International Conference on Learning Representations, (ICLR 2015), Conference Track*, (San Diego, CA, USA), Y. Bengio and Y. LeCun, Eds., May 2015. [Online]. Available: http://arxiv.org/abs/1412.6980.

[109] Richard Baraniuk et al., *Signals and Systems: 3.6: BIBO Stability of Continuous Time Systems*, 2022. [Online]. Available: https://web.archive.org/web/20230824092255/https://eng.libretexts.org/Bookshelves/Electrical_Engineering/Signal_Processing_and_Modeling/Signals_and_Systems_%28Baraniuk_et_al.%29/03%3A_Time_Domain_Analysis_of_Continuous_Time_Systems/3.06%3A_BIBO_Stability_of_Continuous_Time_Systems (visited on 06/28/2022).

[110] B. Rosner, "Percentage Points for a Generalized ESD Many-Outlier Procedure", *Technometrics*, vol. 25, no. 2, pp. 165–172, 1983, ISSN: 00401706. DOI: 10.2307/1268549.

[111] B. Iglewicz and D. C. Hoaglin, *How to detect and handle outliers*, ser. ASQC basic references in quality control. Milwaukee, Wis: ASQC Quality Press, 1993, vol. v. 16, ISBN: 087389247X.

[112] L. Buitinck, G. Louppe, M. Blondel, *et al.*, "API design for machine learning software: Experiences from the scikit-learn project", in *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, (Prague, Czech Republic), Sep. 2013, pp. 108–122.

[113] J. Serrà and J. L. Arcos, "An Empirical Evaluation of Similarity Measures for Time Series Classification", *Knowledge-Based Systems*, vol. 67, pp. 305–314, 2014, ISSN: 09507051. DOI: 10.1016/j.knosys.2014.04.035.

[114] M. Toller, B. C. Geiger, and R. Kern, "A formally robust time series distance metric", *CoRR*, vol. abs/2008.07865, pp. 1–10, 2020. arXiv: 2008.07865.

[115] R. Chalapathy and S. Chawla, "Deep learning for anomaly detection: A survey", *CoRR*, vol. abs/1901.03407, pp. 1–47, 2019. arXiv: 1901.03407.

[116] M. Braei and S. Wagner, "Anomaly detection in univariate time-series: A survey on the state-of-the-art", *CoRR*, vol. abs/2004.00433, p. 39, 2020. arXiv: 2004.00433.

[117] J. Audibert, P. Michiardi, F. Guyard, *et al.*, "USAD: UnSupervised Anomaly Detection on Multivariate Time Series", in *26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '20)*, (Virtual Event, CA, USA), R. Gupta, Y. Liu, J. Tang, *et al.*, Eds., New York, NY, USA: ACM, 2020, pp. 3395–3404. DOI: 10.1145/3394486.3403392.

[118] C. Yin, S. Zhang, J. Wang, *et al.*, "Anomaly Detection Based on Convolutional Recurrent Autoencoder for IoT Time Series", *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pp. 1–11, 2020, ISSN: 2168-2216. DOI: 10.1109/TSMC.2020.2968516.

[119] K. Ding, S. Ding, A. Morozov, *et al.*, "On-Line Error Detection and Mitigation for Time-Series Data of Cyber-Physical Systems using Deep Learning Based Methods", in *15th European Dependable Computing Conference (EDCC'19)*, (Naples, Italy), IEEE, 2019, pp. 7–14, ISBN: 978-1-7281-3929-6. DOI: 10.1109/EDCC.2019.00015.

[120] K. Hundman, V. Constantinou, C. Laporte, *et al.*, "Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding", in *24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD'18)*, (London United Kingdom), ser. KDD '18, London, United Kingdom: ACM, 2018, pp. 387–395, ISBN: 9781450355520. DOI: 10.1145/3219819.3219845.

[121] paolof89, *Time series distance metric*, 2021. [Online]. Available: https://web.archive.org/web/20210507081734/https://stackoverflow.com/%5C%5Cquestions/48497756/time-series-distance-metric (visited on 05/07/2021).

[122] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, *et al.*, "Array programming with NumPy", *Nature*, vol. 585, no. 7825, pp. 357–362, 2020. DOI: 10.1038/s41586-020-2649-2.

[123] C. Jarne, "Simple empirical algorithm to obtain signal envelope in three steps", *CoRR*, vol. abs/1703.06812, 2017. arXiv: 1703.06812.

[124] R. B. Randall and J. Antoni, "Rolling element bearing diagnostics—A tutorial", *Mechanical Systems and Signal Processing*, vol. 25, no. 2, pp. 485–520, 2011, ISSN: 08883270. DOI: 10.1016/j.ymssp.2010.07.017.

[125] J. Grezmak, P. Wang, C. Sun, *et al.*, "Explainable Convolutional Neural Network for Gearbox Fault Diagnosis", *Procedia CIRP*, vol. 80, pp. 476–481, 2019. DOI: 10.1016/j.procir.2018.12.008.

[126] H. Liu, L. Li, and J. Ma, "Rolling Bearing Fault Diagnosis Based on STFT-Deep Learning and Sound Signals", *Shock and Vibration*, vol. 2016, pp. 1–12, 2016, ISSN: 1070-9622. DOI: 10.1155/2016/6127479.

[127] R. Liu, B. Yang, E. Zio, *et al.*, "Artificial intelligence for fault diagnosis of rotating machinery: A review", *Mechanical Systems and Signal Processing*, vol. 108, pp. 33–47, 2018, ISSN: 08883270. DOI: 10.1016/j.ymssp.2018.02.016.

[128] R. Zhao, R. Yan, Z. Chen, *et al.*, "Deep learning and its applications to machine health monitoring", *Mechanical Systems and Signal Processing*, vol. 115, pp. 213–237, 2019, ISSN: 08883270. DOI: 10.1016/j.ymssp.2018.05.050.

[129] Mey, Oliver and Neudeck, Willi and Schneider, Andre and Enge-Rosenblatt, Olaf, "Machine Learning-Based Unbalance Detection of a Rotating Shaft Using Vibration Data", in *25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA2020)*, (Vienna, Austria), IEEE, 2020, pp. 1610–1617, ISBN: 978-1-7281-8956-7. DOI: 10.1109/ETFA46521.2020.9212000.

[130] G. Serin, B. Sener, A. M. Ozbayoglu, *et al.*, "Review of tool condition monitoring in machining and opportunities for deep learning", *The International Journal of Advanced Manufacturing Technology*, vol. 109, no. 3-4, pp. 953–974, 2020, ISSN: 0268-3768. DOI: 10.1007/s00170-020-05449-w.

[131] S. Zhang, S. Zhang, B. Wang, *et al.*, "Deep Learning Algorithms for Bearing Fault Diagnostics—A Comprehensive Review", *IEEE Access*, vol. 8, pp. 29 857–29 881, 2020. DOI: 10.1109/ACCESS.2020.2972859.

[132] O. Janssens, V. Slavkovikj, B. Vervisch, *et al.*, "Convolutional Neural Network Based Fault Detection for Rotating Machinery", *Journal of Sound and Vibration*, vol. 377, pp. 331–345, 2016, ISSN: 0022460X. DOI: 10.1016/j.jsv.2016.05.027.

[133] E. P. Carden and P. Fanning, "Vibration Based Condition Monitoring: A Review", *Structural Health Monitoring*, vol. 3, no. 4, pp. 355–377, 2004, ISSN: 1475-9217. DOI: 10.1177/1475921704047500.

[134] M. Vishwakarma, R. Purohit, V. Harshlata, *et al.*, "Vibration Analysis & Condition Monitoring for Rotating Machines: A Review", *Materials Today: Proceedings*, vol. 4, no. 2, pp. 2659–2664, 2017, ISSN: 22147853. DOI: 10.1016/j.matpr.2017.02.140.

[135] J. T. Renwick and P. E. Babson, "Vibration Analysis—A Proven Technique as a Predictive Maintenance Tool", *IEEE Transactions on Industry Applications*, vol. IA-21, no. 2, pp. 324–332, 1985, ISSN: 0093-9994. DOI: 10.1109/TIA.1985.349652.

[136] D. Kateris, D. Moshou, X.-E. Pantazi, *et al.*, "A machine learning approach for the condition monitoring of rotating machinery", *Journal of Mechanical Science and Technology*, vol. 28, no. 1, pp. 61–71, 2014, ISSN: 1738-494X. DOI: 10.1007/S12206-013-1102-Y.

[137] S. Singh and M. Vishwakarma, "A Review of Vibration Analysis Techniques for Rotating Machines", *International Journal of Engineering Research and Technology*, vol. V4, no. 03, 2015. DOI: 10.17577/IJERTV4IS030823.

[138] O. Mey, A. Schneider, O. Enge-Rosenblatt, *et al.*, "Condition Monitoring of Drive Trains by Data Fusion of Acoustic Emission and Vibration Sensors", *Processes*, vol. 9, no. 7, p. 1108, 2021, Dataset link: https://fordatis.fraunhofer.de/handle/fordatis/151.2. DOI: 10.3390/pr9071108.

[139] Erik Swanson, Chris D. Powell, and Sorin Weissman, "A practical review of rotating machinery critical speeds and modes", *Sound and Vibration*, vol. 39, pp. 10–17, May 2005. [Online]. Available: http://www.sandv.com/downloads/0505swan.pdf.

[140] A. Brandt, *Noise and Vibration Analysis*. Chichester, UK: John Wiley & Sons, Ltd, 2011, ISBN: 9780470978160. DOI: 10.1002/9780470978160.

[141] K. Uchtmann and R. Wirth, "Maschinendiagnose an drehzahlveraenderlichen Antrieben mittels Ordnungsanalyse", *Antriebstechnik*, vol. 38, pp. 44–49, 1999. [Online]. Available: https://maschinendiagnose.de/mosaic/_M_userfiles/PDF/Downloads_DE/Fachbeitraege/ordnungsanalyse.pdf.

[142] Y. Wang, P. W. Tse, B. Tang, *et al.*, "Order spectrogram visualization for rolling bearing fault detection under speed variation conditions", *Mechanical Systems and Signal Processing*, vol. 122, pp. 580–596, 2019, ISSN: 08883270. DOI: 10.1016/j.ymssp.2018.12.037.

[143] O. Mey, W. Neudeck, A. Schneider, *et al.*, *Vibration Measurements on a Rotating Shaft at Di erent Unbalance Strengths*, 2020. DOI: 10.24406/fordatis/65.2.

[144] G. K. Durbhaka and B. Selvaraj, "Predictive maintenance for wind turbine diagnostics using vibration signal analysis based on collaborative recommendation approach", in *2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, (Jaipur, India), IEEE, Sep. 2016, pp. 1839–1842, ISBN: 978-1-5090-2029-4. DOI: 10.1109/ICACCI.2016.7732316.

[145] A. G. Nath, S. S. Udmale, and S. K. Singh, "Role of artificial intelligence in rotor fault diagnosis: a comprehensive review", *Artificial Intelligence Review*, vol. 54, no. 4, pp. 2609–2668, 2021, ISSN: 0269-2821. DOI: 10.1007/s10462-020-09910-w.

[146] Z. Chen and W. Li, "Multisensor Feature Fusion for Bearing Fault Diagnosis Using Sparse Autoencoder and Deep Belief Network", *IEEE Transactions on Instrumentation and Measurement*, vol. 66, no. 7, pp. 1693–1702, 2017, ISSN: 00189456. DOI: 10.1109/TIM.2017.2669947.

[147] W. A. Smith and R. B. Randall, "Rolling element bearing diagnostics using the Case Western Reserve University data: A benchmark study", *Mechanical Systems and Signal Processing*, vol. 64-65, pp. 100–131, 2015, ISSN: 08883270. DOI: 10.1016/j.ymssp.2015.04.021.

[148] D. Wang, "K-nearest neighbors based methods for identification of di erent gear crack levels under di erent motor speeds and loads: Revisited", *Mechanical Systems and Signal Processing*, vol. 70-71, pp. 201–208, 2016, ISSN: 08883270. DOI: 10.1016/j.ymssp.2015.10.007.

[149] T. P. Carvalho, F. A. A. M. N. Soares, R. Vita, *et al.*, "A systematic literature review of machine learning methods applied to predictive maintenance", *Computers & Industrial Engineering*, vol. 137, pp. 1–10, 2019, ISSN: 03608352. DOI: 10.1016/j.cie.2019.106024.

[150] M. J. Hasan, M. Sohaib, and J.-M. Kim, "An Explainable AI-Based Fault Diagnosis Model for Bearings", *Sensors (Basel, Switzerland)*, vol. 21, no. 12, pp. 1–34, 2021. DOI: 10.3390/s21124070.

[151] J. Grezmak, J. Zhang, P. Wang, *et al.*, "Multi-stream convolutional neural network-based fault diagnosis for variable frequency drives in sustainable manufacturing systems", *Procedia Manufacturing*, vol. 43, pp. 511–518, 2020, ISSN: 23519789. DOI: 10.1016/j.promfg.2020.02.181.

[152] A. K. Ovacikli, P. Paajarvi, and J. P. LeBlanc, "Skewness as an objective function for vibration analysis of rolling element bearings", in *8th International Symposium on Image and Signal Processing and Analysis (ISPA 2013)*, IEEE, Sep. 2013, pp. 462–466, ISBN: 978-953-184-194-8. DOI: 10.1109/ISPA.2013.6703785.

[153] H.-Y. Chen and C.-H. Lee, "Vibration Signals Analysis by Explainable Artificial Intelligence (XAI) Approach: Application on Bearing Faults Diagnosis", *IEEE Access*, vol. 8, pp. 134 246–134 256, 2020. DOI: 10.1109/ACCESS.2020.3006491.

[154] J. Luo, S. Zhang, M. Zhong, *et al.*, "Order Spectrum Analysis for Bearing Fault Detection via Joint Application of Synchrosqueezing Transform and Multiscale Chirplet Path Pursuit", *Shock and Vibration*, vol. 2016, pp. 1–11, 2016, ISSN: 1070-9622. DOI: 10.1155/2016/2976389.

[155] S. Io e and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift", *CoRR*, 2015. arXiv: 1502.03167.

[156] O. Mey, W. Neudeck, A. Schneider, *et al.*, *Machine learning-based unbalance detection of a rotating shaft using vibration data*, 2020. arXiv: 2005.12742 [eess.SP].

[157] R. Guidotti, A. Monreale, S. Ruggieri, *et al.*, "A Survey of Methods for Explaining Black Box Models", *ACM Computing Surveys*, vol. 51, no. 5, pp. 1–42, 2019, ISSN: 0360-0300. DOI: 10.1145/3236009.

[158] J. Grezmak, J. Zhang, P. Wang, *et al.*, "Interpretable Convolutional Neural Network Through Layer-wise Relevance Propagation for Machine Fault Diagnosis", *IEEE Sensors Journal*, vol. 20, no. 6, pp. 3172–3181, 2020, ISSN: 1530-437X. DOI: 10.1109/JSEN.2019.2958787.

[159] J. Kim and J.-M. Kim, "Bearing Fault Diagnosis Using Grad-CAM and Acoustic Emission Signals", *Applied Sciences*, vol. 10, no. 6, pp. 1–12, 2020. DOI: 10.3390/app10062050.

[160] C.-J. Lin and J.-Y. Jhang, "Bearing Fault Diagnosis Using a Grad-CAM-Based Convolutional Neuro-Fuzzy Network", *Mathematics*, vol. 9, no. 13, pp. 1–19, 2021. DOI: 10.3390/math9131502.

[161] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why should i trust you?: Explaining the predictions of any classifier", in *22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2016)*, New York, NY, USA: Association for Computing Machinery, 2016, pp. 1135–1144, ISBN: 9781450342322. DOI: 10.1145/2939672.2939778. [Online]. Available: https://doi.org/10.1145/2939672.2939778.

[162] P. Lopes, E. Silva, C. Braga, *et al.*, "XAI Systems Evaluation: A Review of Human and Computer-Centred Methods", *Applied Sciences*, vol. 12, no. 19, p. 9423, 2022. DOI: 10.3390/app12199423.

[163] I. Hameed, S. Sharpe, D. Barcklow, *et al.*, *Based-xai: Breaking ablation studies down for explainable artificial intelligence*, 2022. arXiv: 2207.05566 [cs.LG].

[164] M. Saeki, J. Ogata, M. Murakawa, *et al.*, "Visual explanation of neural network based rotation machinery anomaly detection system", in *IEEE International Conference on Prognostics and Health Management (ICPHM 2019)*, (San Francisco, CA, USA), IEEE, 2019, pp. 1–4, ISBN: 978-1-5386-8357-6. DOI: 10.1109/ICPHM.2019.8819396.

[165] A. Adadi and M. Berrada, "Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI)", *IEEE Access*, vol. 6, no. 6, pp. 52 138–52 160, 2018. DOI: 10.1109/ACCESS.2018.2870052.

[166] J. V. Jeyakumar, J. Noor, Y.-H. Cheng, *et al.*, "How can i explain this to you? an empirical study of deep neural network explanation methods", in *34th International Conference on Neural Information Processing Systems (NIPS 2020)*, (Vancouver, BC, Canada), ser. NIPS'20, Curran Associates Inc., 2020, ISBN: 9781713829546.

[167] Q.-s. Zhang and S.-c. Zhu, "Visual interpretability for deep learning: a survey", *Frontiers of Information Technology & Electronic Engineering*, vol. 19, no. 1, pp. 27–39, 2018, ISSN: 2095-9184. DOI: 10.1631/FITEE.1700808.

[168] A. Das and P. Rad, *Opportunities and Challenges in Explainable Artificial Intelligence XAI: A Survey*, 2020. arXiv: 2006.11371.

[169] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje, "Learning Important Features Through Propagating Activation Di erences", in *34th International Conference on Machine Learning (ICML 2017)*, (Sydney, NSW Australia), ser. Proceedings of Machine Learning Research, vol. 70, PMLR, Aug. 2017, pp. 3145–3153. [Online]. Available: https://proceedings.mlr.press/v70/shrikumar17a.html.

[170] G. Montavon, S. Lapuschkin, A. Binder, *et al.*, "Explaining nonlinear classification decisions with deep Taylor decomposition", *Pattern Recognition*, vol. 65, pp. 211–222, 2017, ISSN: 00313203. DOI: 10.1016/j.patcog.2016.11.008.

[171] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks", in *13th European Conference Computer Vision (ECCV 2014)*, (Zurich, Switzerland), Cham: Springer International Publishing, 2014, pp. 818–833, ISBN: 978-3-319-10590-1. DOI: 10.1007/978-3-319-10590-1_53.

[172] G. Schwalbe and B. Finzel, "A comprehensive taxonomy for explainable artificial intelligence: A systematic survey of surveys on methods and concepts", *Data Mining and Knowledge Discovery*, pp. 1–59, 2023, ISSN: 1384-5810. DOI: 10.1007/s10618-022-00867-8.

[173] S. M. Lundberg and S.-I. Lee, "A Unified Approach to Interpreting Model Predictions", in *31st International Conference on Neural Information Processing Systems (NIPS 2017)*, (Long Beach California, USA), I. Guyon, U. V. Luxburg, S. Bengio, *et al.*, Eds., vol. 30, Curran Associates, Inc, 2017.

[174] Emanuel Metzenthin, *LIME For Time*, *github Repository*. [Online]. Available: https://github.com/emanuel-metzenthin/Lime-For-Time (visited on 07/11/2022).

[175] S. Bach, A. Binder, G. Montavon, *et al.*, "On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation", *PloS one*, vol. 10, no. 7, pp. 1–46, 2015, ISSN: 1932-6203. DOI: 10.1371/journal.pone.0130140.

[176] G. Montavon, A. Binder, S. Lapuschkin, *et al.*, "Layer-Wise Relevance Propagation: An Overview", in *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, W. Samek, G. Montavon, A. Vedaldi, *et al.*, Eds., ser. Springer eBooks Computer Science. Cham: Springer, 2019, vol. 11700, pp. 193–209, ISBN: 978-3-030-28953-9. DOI: 10.1007/978-3-030-28954-6_10.

[177] M. Ancona, E. Ceolini, A. C. Öztireli, *et al.*, *A unified view of gradient-based attribution methods for deep neural networks*, 2017. arXiv: 1711.06104.

[178] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic Attribution for Deep Networks", in *34th International Conference on Machine Learning (ICML 2017)*, (Sydney, NSW Australia), ser. ICML'17, PMLR, 2017, pp. 3319–3328.

[179] D. Smilkov, N. Thorat, B. Kim, *et al.*, *Smoothgrad: Removing noise by adding noise*, 2017. arXiv: 1706.03825.

[180] B. Zhou, A. Khosla, A. Lapedriza, *et al.*, "Learning Deep Features for Discriminative Localization", in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2016)*, (Las Vegas, NV, USA), IEEE, Jul. 2016, pp. 2921–2929, ISBN: 978-1-4673-8851-1. DOI: 10.1109/CVPR.2016.319.

[181] R. R. Selvaraju, M. Cogswell, A. Das, *et al.*, "Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization", *International Journal of Computer Vision*, vol. 128, no. 2, pp. 336–359, 2020, ISSN: 0920-5691. DOI: 10.1007/s11263-019-01228-7.

[182] M. S. Kim, J. P. Yun, and P. Park, "An Explainable Neural Network for Fault Diagnosis With a Frequency Activation Map", *IEEE Access*, vol. 9, pp. 98 962–98 972, 2021. DOI: 10.1109/ACCESS.2021.3095565.

[183] H. Wang, Z. Wang, M. Du, *et al.*, "Score-cam: Score-weighted visual explanations for convolutional neural networks", in *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW 2020)*, (Seattle, WA, USA), IEEE, Jun. 2020, pp. 111–119, ISBN: 978-1-7281-9360-1. DOI: \url{10.1109/CVPRW50498.2020.00020}.

[184] M. S. Kim, J. P. Yun, and P. Park, "An Explainable Convolutional Neural Network for Fault Diagnosis in Linear Motion Guide", *IEEE Transactions on Industrial Informatics*, vol. 17, no. 6, pp. 4036–4045, 2021, ISSN: 1551-3203. DOI: 10.1109/TII.2020.3012989.

[185] U. Schlegel, H. Arnout, M. El-Assady, *et al.*, "Towards A Rigorous Evaluation Of XAI Methods On Time Series", in *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, (Seoul, Korea (South)), IEEE, 2019, pp. 4197–4201, ISBN: 978-1-7281-5023-9. DOI: 10.1109/ICCVW.2019.00516.

[186] Y. Lin, W. Lee, and Z. B. Celik, *What do you see? evaluation of explainable artificial intelligence (XAI) interpretability through neural backdoors*, 2020. arXiv: 2009.10639.

[187] Maximilian Alber, Sebastian Lapuschkin, Philipp Seegerer, *et al.*, "iNNvestigate Neural Networks!", *Journal of Machine Learning Research*, vol. 20, no. 93, pp. 1–8, 2019. [Online]. Available: http://jmlr.org/papers/v20/18-540.html.

[188] O. Mey and D. Neufeld, *Explainable AI Algorithms for Vibration Data-based Fault Detection: Use Case-adadpted Methods and Critical Evaluation*, 2022. [Online]. Available: https://github.com/o-mey/xai-vibration-fault-detection.

[189] H. Kaur, H. Nori, S. Jenkins, *et al.*, "Interpreting Interpretability: Understanding Data Scientists' Use of Interpretability Tools for Machine Learning", in *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, (Honolulu HI USA), R. Bernhaupt, F. '. Mueller, D. Verweij, *et al.*, Eds., New York, NY, USA: ACM, 2020, pp. 1–14, ISBN: 9781450367080. DOI: 10.1145/3313831.3376219.

[190] P.-J. Kindermans, S. Hooker, J. Adebayo, *et al.*, "The (Un)reliability of Saliency Methods", in *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, W. Samek, G. Montavon, A. Vedaldi, *et al.*, Eds., Cham: Springer International Publishing, 2019, pp. 267–280, ISBN: 978-3-030-28954-6. DOI: 10.1007/978-3-030-28954-6_14.

[191] S. Hooker, D. Erhan, P.-J. Kindermans, *et al.*, "A benchmark for interpretability methods in deep neural networks", in *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2019.

[192] *34th International Conference on Machine Learning (ICML 2017)*, (Sydney, NSW Australia), Aug. 2017.