KTH ROYAL INSTITUTE
OF TECHNOLOGY

Degree Project in the Field of Technology and the Main Field of Study Computer science

Second cycle, 30 credits

Anomaly detection for prediction of failures in manufacturing environments

Machine learning based semi-supervised anomaly detection for multivariate time series to predict failures in a CNC-machine

FELIX BOLTSHAUSER

Stockholm, Sweden, 2023

Anomaly detection for prediction of failures in manufacturing environments

Machine learning based semi-supervised anomaly detection for multivariate time series to predict failures in a CNC-machine

FELIX BOLTSHAUSER

Date: June 28, 2023

Supervisor: Erik Fransén
Examiner: Aristides Gionis
School of Electrical Engineering and Computer Science
Host company: Nytt-tech AB
Swedish title: Anomalidetektering för prediktion av fel i tillverkningsmiljöer
Swedish subtitle: Maskininlärningsbaserad delvis övervakad anomalidetektering av multivariata tidsserier för att förutsäga fel i en CNC-maskin

Abstract

For manufacturing enterprises, the potential of collecting large amounts of data from production processes has enabled the usage of machine learning for prediction-based monitoring and maintenance of machines. Yet common maintenance strategies still include reactive handling of machine failures or schedule-based maintenance conducted by experienced personnel. Both of which are time-consuming and costly for manufacturing enterprises. The incorporation of anomaly detection for production processes alleviates several problems connected to these resource-intensive maintenance strategies. Anomaly detection enables real-time maintenance alarms derived from the occurrence of anomalies and thereby a foundation for proactive maintenance during manufacturing. However, to realize this, one needs to investigate the correlation between machine failure and anomalies in the data. For the machine learning models, it is also of essence to handle the imbalance between failure and normal working condition data. In this work, we investigate the potential of anomaly detection to predict future tool failures of an active CNC-machine based on multivariate time series data collected through the standardized data collection protocol MTConnect. Two semi-supervised anomaly detection methods, DeepAnT and ROCKET OCSVM, were tested. Training and evaluation of the two models were conducted on three production part processes and the difference in anomaly distribution previous to failure and in the normal machine working condition was investigated. The results showed that both models, for all the investigated tool failures belonging to the three production part processes, found an abundance of anomalies preceding failure when compared to the normal working condition of the machines. For certain tool failures, the anomalies were found as far back as seven production cycles before failure, while other anomalies were mainly uncovered close to the failure. Furthermore, it was shown that both models performed optimally with 100 production cycles before tool failures excluded from training, indicating that more anomalies further back connected to failure or possible long-term degradation of machine tools could exist. Lastly, ROCKET OCSVM with RBF kernel showed greater reliability compared to the DeepAnT method in separating the normal working condition data of the CNC machine against the pre-failure data based on anomaly distribution. In conclusion, anomaly detection shows promising results in indicating future machine failure and could serve as a foundation for proactive maintenance strategies of machines. By incorporating proactive strategies, machine downtime, operator maintenance time, and resources and expenses resulting

from machine failure could be reduced.

Keywords

Machine learning, Anomaly Detection, DeepAnT, ROCKET, OCSVM, manufacturing, predictive maintenance

Sammanfattning

För produktionsföretag har potentialen att samla in stora mängder data från produktionsprocesser möjliggjort användningen av prediktionsbaserad övervakning och underhåll av maskiner genom maskininlärning. Ändå så utgörs fortfarande vanliga underhållsstrategier av reaktiv hantering av maskinfel eller schema baserat underhåll som utförs av erfaren personal. Båda dessa är tidskrävande och kostsamma för tillverkningsföretag. Införandet av anomali detektering för produktionsprocesser lindrar flera problem kopplade till dessa resursintensiva underhållsstrategier. Det möjliggör underhålls-larm i realtid härledda från förekomsten av anomalier, vilket skapar en grund för proaktivt underhåll under tillverkningen. Men för att möjliggöra detta måste man undersöka sambandet mellan maskinfel och anomalier i data utifrån definierade insamlingsmetod. Det är också viktigt att hantera obalansen mellan fel och normal arbetstillstånd data för maskininlärningsmodellerna. I det här arbetet undersöker vi potentialen för delvis övervakad anomali detektering för att förutsäga framtida verktyg fel hos en aktiv CNC-maskin baserat på multivariat tidsseriedata som samlats in genom det standardiserade datainsamling protokollet MT Connect. Två anomali detekterings metoder som endast tränats på normala arbetsförhållanden för maskiner testades, DeepAnT och ROCKET OCSVM. Träning och utvärdering av de två modellerna genomfördes på tre produktionsdelprocesser och skillnaden i anomali fördelning före fel och i det normala maskinens arbetstillstånd undersöktes. Resultaten visade att båda modellerna, för alla undersökta verktygsfel som hör till de tre produktionsdelprocesserna, fann ett överflöd av anomalier före fel i jämförelse med maskinernas normala arbetstillstånd. För vissa verktygsfel hittades anomalierna så långt tillbaka som sju produktionscykler före fel, medan andra anomalier huvudsakligen upptäcktes nära felet. Vidare visades det att båda modellerna presterar optimalt med 100 produktionscykler före verktygsfel uteslutna från träningen, vilket tyder på att fler anomalier tidigare än de åtta produktions cyklarna undersökta innan fel eller eventuell långvarig försämring av verktygsmaskiner kan förekomma. Slutligen visade ROCKET OCSVM med RBF som kärnfunktion större tillförlitlighet i jämförelse med DeepAnT metoden gällande att separera CNC-maskinens normala arbetstillstånd data från pre-failure-data baserat på anomali fördelning. Sammanfattningsvis visar avvikelse detektering lovande resultat för att indikera framtida maskinfel och kan fungera som en grund för proaktivt underhåll av maskiner. Genom att införskaffa proaktiva strategier kan maskinernas stilleståndstid, operatörens underhållstid samt resurser och

kostnader till följd av maskinfel minskas.

Nyckelord

Maskin inlärning, Anomali Detektion, DeepAnT, ROCKET, OCSVM, tillverkning, prediktivt underhåll

Acknowledgments

I would like to thank my KTH supervisor Erik Fransén for his continued assistance regarding technical and structural advice in this research. I would also like to extend my gratitude to my supervisor at the company of Nytt, Praveen Natarajan, and the CEO at the company, Sture Wikman, for providing me the opportunity of conducting this thesis at the company of Nytt.

Stockholm, June 2023
Felix Boltshauser

Contents

List of Figures

List of Tables

List of acronyms and abbreviations

ADAM	adaptive moment estimation
CNN	convolutional neural network
MAE	mean absolute error
MLP	multi layer perceptron
OCSVM	one class support vector machine
ROCKET	random convolutional kernel transform
SVM	support vector machine
t-SNE	t-Distributed Stochastic Neighbor Embedding

Chapter 1

Introduction

Machine learning has seen a wide variety of use cases in industrial settings during the later years [1], including performance predictions [2] online quality controls [3], defect and anomaly detection [4], among many others. The increase in use cases is largely due to the big data revolution, the large amount of available data following industry 4.0 with the widespread increase in sensory data collection techniques [5] have resulted in machine learning algorithms seeing both an increase in applications and performance.

For manufacturing, following the increased complexity of production processes, non-automated maintenance strategies enabled by human intervention are generally difficult to realize. Furthermore, the most common maintenance strategies used today have been shown to decrease the overall productivity of manufacturing equipment by five to twenty percent [5]. This is because of several unwanted consequences following poor maintenance, including increased downtime, machine failure, and other costly repercussions. It has been shown that the consequent loss of time followed by machine failure costs manufacturing enterprises around $50 billion each year [5]. It is therefore necessary for manufacturing organizations to leverage the usage of machine learning to counteract these complications and stay competitive in the market. This has led to the emerging interest in predictive maintenance, a cross-interdisciplinary field between data analytics and manufacturing maintenance. It can be defined as any proactive maintenance strategies that utilize data analytics and machine learning to monitor equipment health to address potential future failures [6]. Several surveys have discussed its positive effects on production processes [7], [8] and it has been shown to achieve a ten times return rate of investments [9] further supporting the integration

of machine learning models into the maintenance pipeline of manufacturing organizations.

For predictive maintenance, the first step [10] in automating and streamlining the maintenance process of manufacturing machines is anomaly detection [11]. Anomaly detection aims towards investigating abnormal patterns that do not belong in the normal distribution of the investigated data [5] and which are correlated or causally linked to later machine failures, and thereby can enable automatic alarm systems with the intent of indicating future machine failure.

The potential of using anomaly detection to indicate future failures of a CNC machine monitored by the company of Nytt will be investigated in this research. This, in an attempt to widen the knowledge about anomaly detection in manufacturing settings and reflect on the potential of improving the current maintenance strategies conducted at Nytt.

1.1 Context

Nytt is a company that developed a dashboard interface based on time series data collected continuously through a data collection platform for manufacturing companies [12]. The machine monitoring system developed by the company of Nytt was created to aid manufacturing companies in becoming more data-driven and enable better communication between operators, managers, and machines. One of the data-gathering processes follows the MTConnect protocol [13], which is a standardized data vocabulary used for collecting manufacturing machine data. Through this information, companies can follow a bottom-up approach in their decision-making, where statistics regarding the data can be followed in a dashboard and help operators make informed decisions while sharing knowledge with managers. The system has been used by multiple small to medium-sized manufacturing enterprises and created a foundation to improve strategies, productivity, and resource usage.

Currently, the data collection methodology is active at Precima enterprise [14], where it monitors a CNC-machine. CNC-machines, or computer numerical control (CNC) machine, are a commonly occurring type of manufacturing machine that relies on computer-aided assistance, or computer-aided design draftings (CADs), of each production part produced by the

machine. These computer programs control machine tools for the sequential removal of layers from raw material work-pieces to produce specific parts [15]. Because of the material removal, the process is generally referred to as subtractive manufacturing and is commonly occurring in industries such as telecommunication, and aerospace [15], and for the investigated machine at Precima, drains for the sewage industry. Tools controlled by the CAD programs are grouped into three different parts; drilling operations, milling operations, and turning operations. Drilling operations involve perpendicular or angular feeding of raw material onto drilling tools. Milling operations involve rotational and dual-point cutting of raw material. Turning operations involve positioning the raw material in the 3D space for the drilling and milling operations [15]. It is important to get a basic understanding of these operations since maintenance work on the machine is generally tool-based and data collection processes like MTConnect focus on features such as load [16], positioning, vibration, and acoustics [17] of the different machine tools.
While the current system has been shown to increase manufacturing productivity by twenty to thirty percent, there is still potential for growth. Nytt wants to extend its services by incorporating automatic monitoring of machine health and thereby build maintenance strategies that are independent of human intervention. They want to do so by using the historical data of the Precima CNC-machine to investigate the possibility of realizing a proactive maintenance strategy.

1.2 Problem

1.2.1 Problem and definition

This thesis was proposed by the company of Nytt, motivated by the rapid growth in predictive maintenance services. Nytt wanted to investigate the possibility of using their standardized data to enable indications of future machine failures. Indeed, the potential of using the collected data to predict future failures in the machines has not been fully explored and it is of interest to investigate the data characteristics and draw conclusions regarding the possible correlation between data patterns and machine failures. Predictive maintenance is indeed a necessity for companies to remain competitive in today's market since it brings several crucial benefits to manufacturing enterprises, including increased equipment lifetime, reduction of the maintenance cost, decreased downtime, safety improvements for operators, and improved reliability of machines [18].

Additionally, the efficiency of maintenance strategies in manufacturing processes today is in many cases inadequate following the use of resource-intensive maintenance strategies only conducted by experienced personnel. Using human judgment is inefficient considering the usual delays that could lead to consequent downtime of machines.
To build a foundation for predictive maintenance, anomaly detection is usually the first step since it can enable failure indication at early stages and maintenance can be conducted without unplanned downtime [19]. The field of anomaly detection also has several future challenges presented in Section 1.2.2 that need to be addressed. Following the need to combat these, as well as providing a foundation of predictive maintenance for the company of Nytt, two semi-supervised anomaly detection methods, DeepAnT [20] and ROCKET OCSVM, will be investigated. The research then aims to answer the following research questions:

1. *How well does the DeepAnT model forecast the multivariate time series features of a manufacturing machine dependent on different parameter setups?*

2. *How well does DeepAnT separate the normal working states and anomaly states leading up to failures of machines dependent on the anomaly distribution?*

3. *How well does ROCKET OCSVM separate the normal working states and anomaly states leading up to failures of machines dependent on the anomaly distribution?*

4. *How early before a failure do the two models, DeepAnT and ROCKET OCSVM find anomalous behavior that could be linked to the failures?*

5. *How does DeepAnT and ROCKET OCSVM differ in separating the normal working states and anomaly states leading up to failures of machines? Additionally, which seems to be the most reliable for manufacturing enterprises?*

6. *To avoid training the semi-supervised models for the class of normal operation data on anomalous data, what effect does the removal of different time windows previous to failures have on the anomaly detection task? What time window size preceding machine failure should be removed to have all anomalous data removed connected to failures?*

1.2.2 Scientific and engineering issues

The usage of anomaly detection in monitoring IoT data from sensors has several future challenges, as described by Stojanovic et al. [11]. In real manufacturing setups, the data distribution between normal and failure state data is usually unbalanced. This is indeed the case for the data collected at the company of Nytt and there is a need to investigate semi-supervised approaches where models are trained only on only non-anomolous normal state data. They also state that there is a need to further investigate handling multivariate sensor data through model selection.

The second problem, which was presented by Villa-Pérez et al., states that the usage of semi-supervised learning needs further research regarding the impact of parameter tuning for current models [21]. Several hyper-parameters that could be of interest but have not been thoroughly investigated are the temporal window size for the separation of normal and failure data in semi-supervised models, as well as the anomaly distribution dependent on the time frame before failures in machines. Indeed, as stated by Ruff et al. the main limitation for semi-supervised LUPE (see section 2.4.2) is the unclear separation of normal and failure data [22].

Lastly, the proposed model DeepAnt has shown to perform well for multivariate time-series anomaly detection tasks based on numerous data sets from global cloud enterprises [23], as well as in the Yahoo Webscope and Numenta Anomaly Benchmark (NAB) data sets [24]. However, in the original article [20], the manufacturing sector was mentioned as a field that could benefit from the usage of the model, but no further research could be found that has investigated its potential in an actual manufacturing environment. The original article also states that future research should investigate the usage of pre-processing techniques to investigate its effect on the forecasting component of the model.
For the second model, while ROCKET is currently a state-of-the-art technique used for time series classification, the availability of work for its usage in anomaly detection in time series is sparse. It has however been used for health monitoring of machines and been shown to outperform previous methodologies [25]. Thereby, there is a need to present its usability in the area of time series anomaly detection.

1.3 Purpose

The reason for incorporating anomaly detection to indicate future failures is because it enables the company of Nytt to aid customer enterprises enhance productivity and limiting expenses and resources. From a societal perspective, as explained by Bahrudin & Selver, predictive maintenance in manufacturing is of great interest in the industry since it could aid organizations in saving money and resources [26]. This is due to the ability to predict, and thereby reduce, machine failure and downtime. This is also supported by Hsieh et al. who states that detecting anomalous behavior of machines in the production line enables companies to reduce the consequent cost of downtime since failures connected to anomalies can be detected at early stages [19]. To enable this, anomalous behavior preceding machine breakdown must be analyzed and separated from the normal machine working state, which is the purpose of the thesis. This includes both if there is an abundance of anomalies preceding failures and if the anomalies have different characteristics than possible normal working state anomalies.

From a research perspective, as described in [27], the majority of the process-level built AI models used for monitoring machines are conducted in a laboratory environment. There is a need to promote AI:s in manufacturing industry settings to facilitate acceptance of AIs.

Furthermore, as presented in Section 1.2.2, there is a need to investigate the handling of unbalanced data through semi-supervised methods, as well as multivariate sensory data. Both these factors are taken into consideration following the model selection. Both DeepANT and ROCKET OCSVM is tested in a semi-supervised manner and can handle multivariate time-series data. Additionally, a pre-processing method is proposed and tested for both of the models. As stated in Section 1.2.2, this was labeled a future research direction by the original authors of the DeepAnT model.

Lastly, the lacking hyper-parameter investigations done for semi-supervised learning models was highlighted in Section 1.2.2. To extend this, we will first investigate how far-back anomalies connected to failures exist. This is also of interest to manufacturing enterprises since it can give insights into the reactive time-window available to handle possible future failures. Secondly, to explore the separation of normal working conditions and failure data, we will treat the temporal window size for separating failure and normal state data as

a hyper-parameter during training. From a research perspective, this could be an efficient solution to the unclear separation between normal and abnormal states for semi-supervised models. For manufacturing enterprises, this can both serve as a guideline for further model training and provide additional information regarding how far back before failure possible degradation or other anomalous pattern exist.

1.4 Goals

As mentioned, the goal of this project is to give the company of Nytt further insights into anomalous behavior of collected data related to machine failure while also investigating several future challenges of anomaly detection. To enable this we have divided the work into the following sub-goals:

- Subgoal 1: Propose a standardized data pre-processing methodology that can aid the company in making the data more compatible with machine learning models. Subsequently, divide the data into normal and failure states based on a defined time window connected to failures.
- Subgoal 2: Develop the semi-supervised anomaly detection models, DeepAnT and ROCKET OCSVM.
- Subgoal 3: Train the models on the normal state data and investigate the effects of different hyper-parameters, including both model-specific parameters and non-model-specific parameters such as the temporal window size needed to separate data belonging to failure states from normal states. For DeepAnT, investigate the forecasting ability by monitoring the continued and final training loss.
- Subgoal 4: Evaluate the models individually on the normal and failure state data to conclude if there is separation in the distribution of anomalies. Furthermore, investigate the characteristics of anomalies preceding failures, including how early before failure possible anomalies appear.
- Subgoal 5: Compare the evaluation results of the two models and discuss similarities and differences. Lastly, consider the benefits and disadvantages of the two architectures and determine which of the two models that are most reliable for manufacturing enterprises.

1.5 Research Methodology

The research methodology presented follows a data-driven approach. Considering the large amount of data available following the emergence of industry 4.0, data-driven approaches are necessary to take advantage of available information [28]. Data-driven approaches in manufacturing often follow empirical methods, where one tries to derive and analyze data based on direct or indirect observations [29]. This follows an inductive process where we will try to infer general rules on the characteristics of the machine data based on the collected observations.
To evaluate the results and draw conclusions, the imbalance of the data has to be taken into consideration through the evaluation metrics and visualization methods of the result.
Lastly, the methodology will also rely on the assumption that machine failures do have anomalies in close proximity, this is an assumption that has to be further discussed through the results.

1.6 Ethics and Sustainability

Several consequences connected to the incorporation of predictive maintenance, including reduction of the maintenance cost, safety improvements for operators, and increased efficiency for resource usage, bring sustainability benefits to society. For instance, the reduced maintenance cost can be connected to the UN sustainable development goal 8.2 through increased economic productivity following technological advancements, safety improvements for operators can be connected to the UN sustainable development goal 8.8 concerning increased workplace safety, and efficient resource usage can be linked to the UN sustainable development goal 8.4 concerning reduced material footprint [30]. Nonetheless, one ethical consideration involves the automation of human working tasks. It is important to note that the model designed in this study will not be used to replace human workers but instead aid them in their tasks. As pointed out by Pavol et al, along with the changing industry 4.0 there is a need to incorporate AI for efficiency, not by replacing humans, but by supporting decision-making in the manufacturing industry [31].

1.7 Delimitations

The project will only use the available data from one manufacturing machine. Since different machines can follow highly different feature patterns, producing a generalized model would be a more advanced task. Thereby we do not investigate the generalized performance of different manufacturing machines in the industry. However, considering that the MTconnect protocol is well established for machine monitoring [32], the methodology can provide future recommendations for other researchers. It is also worth stating that the algorithm will simply be used and tested on previously collected data. Testing the reliability of data collected in real-time is not included in the scope of the research.

1.8 Structure of the thesis

The report follows the given disposition: Chapter 2 presents relevant background information with related work to provide the knowledge needed to understand the methodology. Chapter 3 presents further mathematical background behind the investigated machine learning models, feature extraction methods, and evaluation methodologies. Chapter 4 introduces the data-set, the adaption of the data to the models, and the implementation of the anomaly detection models with the evaluation strategy used to answer the research questions. Chapter 5 presents the results following the proposed methodology and Chapter 6 presents a discussion of the results with limitations, conclusion and future directions to extend the research.

Chapter 2

Background

This chapter will present the theoretical background needed to understand the proposed methodology of this research. In Section 2.1, we will discuss the current maintenance strategies with disadvantages compared to predictive strategies. In Section 2.2, we will describe the data type available for data analytic on the CNC machines, namely time series. In Section 2.3, we will present an overview of machine learning and then present its relationship to anomaly detection in Section 2.4. Lastly, In Section 2.5, we will define feature extraction and the importance of feature construction and selection for predictive models.

2.1 Maintenance in manufacturing

Effective maintenance is considered strategies in manufacturing that can minimize equipment failure, improve equipment condition, and prolong the lifetime of involved tools used in the production process [33]. There exist several groups of maintenance strategies listed below that, following Figure 2.1, fulfill the aforementioned criteria to different degrees

- **Run to failure/reactive maintenance:** This methodology is considered the least reliable and involves conducting maintenance work only when equipment failure has already happened [34]. The resulting downtime and inefficiency of machines are usually compromised following this strategy since handling failures generally requires more extensive work compared to preventive work.

- **Preventive maintenance:** This methodology involves scheduling-based maintenance where the goal is to anticipate failures and counteract

failures followed by indications seen at the scheduled time of monitoring [35]. The downside to this is the possible redundant maintenance time needed by operators that monitors machines, which is both costly and impractical.

- **Condition-based Maintenance:** This methodology involves conducting equipment monitoring when indications of possible performance degradation are present [33]. The indication can be through both physical observations or through data collection of the machines.
- **Predictive maintenance:** This methodology seeks to prevent equipment failure through monitoring and analytics of machine data with the use of machine learning. It aims to detect deviations from normal working conditions and schedule maintenance based on high deviations [36]. This method is generally considered superior in terms of reducing downtime and increasing overall equipment effectiveness on account of the ability to use machine data to make predictions at an early stage of degradation. It also does not require operator intervention which is efficient but creates a reliability problem since to many false alarms could decrease the credibility of the system [37].

Predictive maintenance in itself constitutes a large amount of data analysis techniques, the majority of which rely on the area of anomaly detection [5]. Anomaly detection creates a foundation for alert rules that can be monitored by operators conducting maintenance since anomalies observed in real-time can be used as degradation or failure identification [5]. Note that this process is entirely dependent on the availability of sensory machine data collected over production time. Next, we will present the most commonly available type of data for monitored CNC machines, time series.

2.2 Time series data

Time series data relies on features gathered through continuous measurements over time. The data can be described as an ordered sequence of features at different timestamps that, most commonly, are collected with a constant frequency. Time series are easy to obtain and common in many fields including weather data, stock market analysis [38], or, as in this case, machine manufacturing sensory data. Consider the following example

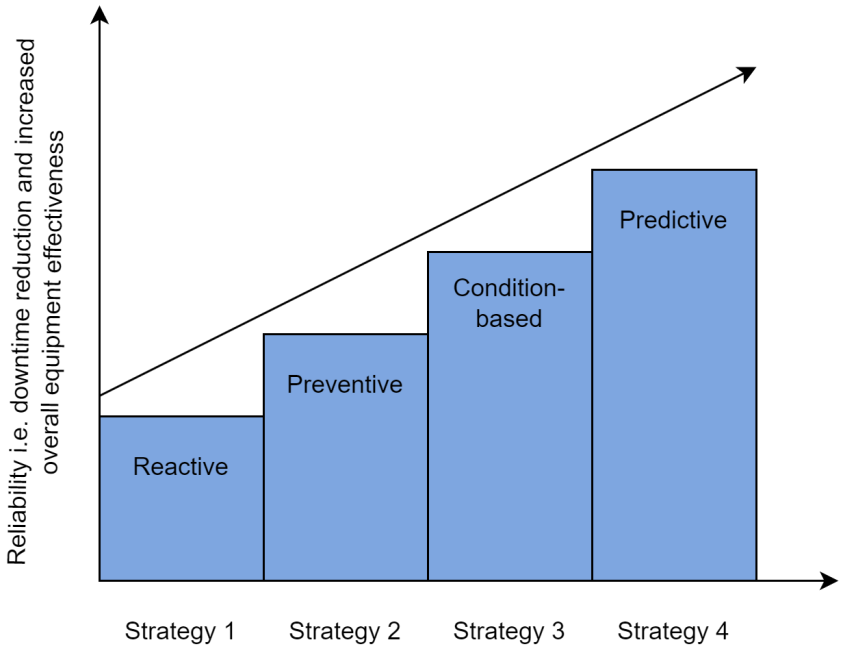


Figure 2.1: Summary of dependency between maintenance strategies and reliability.

$$\{x_1, x_2, x_3 ... x_t\}$$

Each measurement $x_i \in \mathbb{R}^d$ for $i \in [1...t]$ consists of d features that have been collected at the time i. If $d = 1$, then it is referred to as a univariate time series and if $d > 1$, then it is referred to as a multivariate time series. However, this definition of time series does not consider temporal dependencies. Temporal dependencies include factors such as seasonality, which describes the repeating cycles of the features over time, and trend, which describes the decrease or increases in feature mean values over time [39]. If the features measured over time do not have these temporal dependencies, then the time series is said to be stationary.

Following the definition of time series, we will now present what constitutes an anomaly in this specific datatype.

2.2.1 Time series anomalies

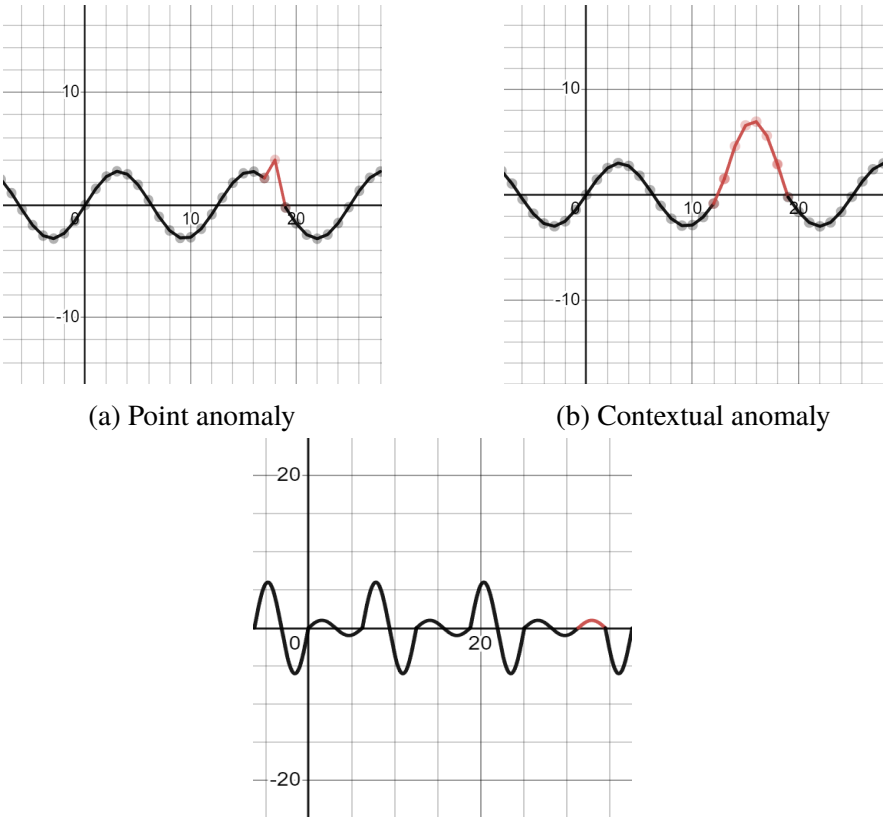
Anomalies can be described as unusual behavior that does not follow the expected pattern in data. When working with time series data, the anomalies are generally categorized into three different types; point anomalies, contextual anomalies, and collective anomalies [40].

- **Point anomalies:** These are single observations in the time series data that deviate from the expected distribution without considering any other input data [40]. Since the context of the data point is not needed, these anomalies are not temporally dependent. An example of a point anomaly is shown in Figure 2.2a.
- **Contextual anomalies:** These are data points that can only be considered anomalous through the context of the data points surrounding them. Thereby, to detect these anomalies, any anomaly detector needs to consider the temporal dependency of the measurements by observing an entire sequence of data points [40]. An example of a contextual anomaly is shown in Figure 2.2b.
- **collective anomalies:** Lastly, collective anomalies differ from point and contextual anomalies through the fact that only a group of time series data points can be considered a collective anomaly. Here, a set of time-series points can be considered anomalous, even though no single data observations are considered to be either a point or contextual anomaly [40]. An example of a contextual anomaly is shown in Figure 2.2c.

The process of detecting and labeling the aforementioned types of anomalies is called anomaly detection. Before presenting this field more in depth, we have to present an overview of the area it belongs to, namely machine learning.

2.3 Machine learning

Machine learning is the process in which computers learn through experience using computational methods [41]. Experiences, in this context, refers to the presence of data describing real-world phenomenon and observations. Machine learning tasks involve building models holding learning algorithms deduced from the experience data being fed to the model during the training



(a) Point anomaly

(b) Contextual anomaly

(c) Collective anomaly

Figure 2.2: Anomaly types in time series data, note that the x-axis displays the observed values y for each time stamp x. The red segments shows the anomaly of the time series.

phase. The process of deducing these learning algorithms can differ, however, machine learning models are generally grouped into the following two categories; *supervised* and *unsupervised*. The difference between the two relies on the presence of labels in the data. Supervised machine learning models try to learn from predefined output characteristics, more formally called labels, related to the input data [42]. The training phase and consequent deduction of learning algorithms involves predicting the label related to each of the input data points. In the unsupervised setup, labels are not present during training, the models instead try to perform pattern recognition through observation of unlabeled data points [42].

Machine learning models can be further grouped on the output characteristics of the data. When a machine learning models provide discrete output values as predictions, it's called a classification model. When the output predictions

are continuous, it's called a regression model [41]. To provide examples, classifications models are well suited in several medical applications, for example in labeling medical images into categorical values based on presence of diseases, such as benign or malignant cancer tumors [43]. Regression models are popular for forecasting applications, for example in predicting future energy consumption based on previous observations and patterns in time series [44].

Lastly, considering that labeled instances usually require heavy human intervention to acquire, which can be time-consuming and expensive, a third group called semi-supervised learning has gained big interest over the later years. These models tries to learn through data sets where only a few labeled instances are defined along with an excessive amount of unlabeled data [45]. In real world applications, where unlabeled data is easily obtained, semi-supervised machine learning algorithms have shown promising results in boosting the performance of classifiers.

2.4 Time series anomaly detection

Given the definitions of anomalies in time-series presented in Section 2.2.1, anomaly detection algorithms try to mark anomalous segments or points in the time series data using machine learning models. A wide variety of algorithms have been proposed for both univariate and multivariate time series in unsupervised, supervised, and semi-supervised setups and will be presented further in Section 2.6.1. Before doing this, we will extend the definitions of unsupervised, supervised, and semi-supervised learning to the anomaly detection task.

2.4.1 Unsupervised and supervised anomaly classification

In unsupervised anomaly detection, there is usually an assumption made that most of the data do not include anomalies. Following this assumption, the training is conducted on the entire data set where the anomaly detection model tries to find a compact distribution of the normal samples and label any data point or sequence that deviates from the distribution as anomalous. The downside of this approach is the bad generalization of the boundary between normal and anomalous data depending on the distribution between normal and

anomalous samples [22]. The assumption of a low amount of anomalies can indeed not always be guaranteed.

In the supervised classification setting classifiers tries to create a maximal separating hyperplane between labeled anomalous and normal samples [22]. The supervised classification task is not always optimal considering the small number of positive labels that are usually present in a lot of real-world anomaly data sets. As previously stated, labeling is a time-consuming task that requires extensive domain knowledge. This leads to the alternative approach of a semi-supervised anomaly detection setup.

2.4.2 Semi-supervised anomaly detection

In anomaly detection, semi-supervised methods involve two alternative approaches separate from the unsupervised and supervised setup. Either only labeled non-anomalous instances are used for training, and labeled anomalies are excluded. Then, the goal is to build a model for the normal class and use the model for the detection of anomalies in the abnormal class during testing [45]. This is more generally titled learning from positive and unlabeled examples (LUPE) [22] and generalized semi-supervised settings where only a few labeled anomalies are present and exploited during training are less common and often specific to the involved domain [46].

2.5 Feature extraction

A key consideration for anomaly detection tasks is feature extraction. Feature extraction aims to transform defined input data into a new feature space to capture the most relevant characteristics of the data for the involved anomaly detection models. What defines a suitable format is often domain-specific and can even require domain knowledge to derive. In either case, this process frequently involves feature construction or feature selection [47], both of which will be presented here.

2.5.1 Feature construction

Feature construction involves taking raw data through a defined pre-processing step to create a new feature space with either the same, an increased, or decreased dimensionality [47]. Prominent examples used in time series anomaly tasks that keep dimensionality include input standardization, which

involves scaling the input features to a zero mean and unit variance, normalization, which involves scaling the input features to a given range, and wavelet transforms [48], which involves decomposing input signals into frequency components that can be analyzed for local features or variations in the data. Another popular method that reduces dimensionality is principal component analysis (PCA)[49], an algorithm used to reduce the dimensionality of data while retaining the majority of the variance. This involves extracting principal components, which are directions in a subspace of the original data.

2.5.2 Feature selection

It is important not to confuse feature construction with feature selection, which is generally done afterward to reduce the input dimensionality. Indeed, feature selection tries to limit the number of input variables to the most discriminate for the involved predictive model [47], which, in the case of anomaly detection, involves finding the input channels that contribute more to identifying anomalies. This is indeed an important part of the model-building pipeline considering the curse of dimensionality. Not only do we want to limit the number of features because of this phenomenon, but feature selection wants to do so by removing redundant information and thereby achieving higher predictive performance with lower computational time [50].

Methods for doing so firstly include filtering, which involves creating a feature rank in the order of how discriminant the features are [47]. An example of this includes correlation coefficients, which have been used to rank input features based on interdependence for predicting the target value in anomaly detection tasks previously [51]. Secondly, we have wrapper techniques, which involve training predictive models on different feature combinations and evaluating the performance. Examples of this include Recursive feature elimination [52]. Thirdly, we have embedded methods, which incorporate the feature selection process into the model training phase using, for example, regularization methods such as LASSO [53].

2.6 Related work

Following the theory behind time series data and anomaly detection using machine learning, we will now present an overview of machine learning methods used for anomaly detection tasks. This is to give an overview of

previous and current state-of-the-art methods used in both general time series anomaly detection, but also anomaly detection applied in the manufacturing industry. It is worth stating that this is by no means a comprehensive summary but simply presents an overview to get a fundamental understanding of the concept of anomaly detection on time series.

2.6.1 Time series anomaly detection

Recent advances in data collection techniques have enabled many fields to collect data in the form of time series. Consequent mining of the data properties has become an important field of research, for example in investigating possible anomalies in the data. Anomalies can both constitute unwanted patterns that we want to clean from the data sets or present events of interest that we want to analyze. In either case, outlier detection handles both situations and has therefore seen significant growth over the last couple of years [54]. Early methods included estimation models that investigated the presence of point anomalies in univariate time series. These methods follow an assumption that the data is generated from one specific process and that deviations from the normal pattern generated from the process should be labeled as anomalous. The earliest methods used piece-wise constant values where anomalies were labeled based on time-window median values or full time-series median values [55]. Other more recent approaches included error measures from the true time series values against the exponential weighted moving average (EWMA) [56], B-splines [57], Gaussian Mixture Models [58], Time series decompositions [59] among others. In essence, they all rely on the following formula

$$\delta < f(|x_{pred} - x_{true}|)$$

where, x_{pred} is the predicted observation at time t and x_{true} is the true value at time t, to label points as anomalies. Indeed, separate methods have been proposed to investigate the error measure through the function f, for example, using human intervention, extreme value theory or Extreme Studentized Deviate [54].

The shortcoming of many of these methods includes their inefficiency to predict anomalies in newly collected data. Which is practical in real-world applications where streaming data is collected in real-time. Therefore, in contrast to these estimation-based methodologies, more novel approaches have

included prediction models such as convolutional neural networks (CNNs) as done by Munir et al.[20] or Recurrent Neural Networks (RNNs) such as Long-Short-Term-Memory (LSTMs), as done by Hundman et al. [60]. These use past observations to make predictions on forthcoming data and can thus label new points as anomalous, based on an error threshold, in real-time.
Both the estimation-based methods and the prediction models use error thresholds of prediction errors to label points as anomalies. However, density-based models have also been investigated, where we define a neighbor criteria function d, with threshold R for all data points x and investigate the number of neighbors δ for time point x_t as follows [54]

$$\delta < x \in X | d(x, x_t) \leq R|$$

To consider the temporal dependency, previous research has tested the methodology on sliding windows [61], and some common methodologies for the neighbor criteria function include local outlier factor (LOF) or density-based spatial clustering (DBSCAN) [62] .

All the aforementioned techniques are viable alternatives to detect point anomalies in univariate time series. However, many real-world data sets consist of multivariate time series. In addition, anomalies are not always observed over time series points, but instead over sequences in the case of collective anomalies. The error criteria are then usually rephrased as follows

$$\delta < f(\sum_{i=n}^{n+p-1} |x_i - \hat{x}_i|))$$

Where n is the time of the first observation in the sequence, p is the sequence length and $\hat{x}_i \in \mathbb{R}^+$ is the predicted value of $x_i \in \mathbb{R}^+$ that belongs to a sequence of length $p - 1$. Numerous models exist to deal with the given setup. Indeed, models that allows for multi-step predictions have has been shown to outperform single-step predictions predictions in regards to capturing changes in amplitude, frequency, or other characteristics connected to collective anomalies [63]. The methods involved in dealing with this more general setup can be said to belong to one out of six families of methods [64]; forecasting, for example, DeepAnT [20] or LSTMAD [65], reconstruction, for example, LSTM-VAE [66], distance, for example, LOF [67], encoding, for example, Ensemble GI [68], distribution, for example, COPOD [69], and tree-based methods, for example, Isolation Forest [70]. Each of these can be further separated based on if they are supervised, unsupervised, or semi-supervised.

2.6.2 Anomaly detection in manufacturing

There has been a wide variety of anomaly detection techniques applied in manufacturing setups. For example, Abdelrahman & Keikhosrokiani used an inspection process to collect positional data of assembly line machines to evaluate several traditional anomaly detection algorithms, including HBOS, IForest, KNN, CBLOF, OCSVM, LOF, and ABOD [71]. Furthermore, Aydemir & Acar investigated using anomaly detection for the prediction of remaining useful life in the turbofan engine degradation dataset [72], a commonly used dataset for benchmarking health monitoring algorithms [73]. Other approaches seen in real-world manufacturing settings include the work of Zhang et al. who combined a feature extraction methodology using PCA with DBSCAN to detect anomalies in time series data extracted from CNC machines. The results showed promising results in the detection of injected anomalies in a real manufacturing setting [74] .

Several deep learning methodologies have also been explored for anomaly detection in manufacturing. For example, Oh & Yun investigated the usage of auto-encoder to separate normal from anomalous sound data in manufacturing machines [75]. The results showed that it was possible to detect anomalies close to machine failure.

Continuing, due to the nature of the data in industrial machines, many of the supervised anomaly detection algorithms become futile. The majority of the data collected from manufacturing machines are usually collected unlabeled or with a small portion of labeled data, for example, due to failures. Consequently, there has been an increase in semi-supervised machine learning anomaly detection methods investigated during the last couple of years, for example, using auto-encoders [76], including the works of Liu et al. who used CNN-based auto-encoders to detect surface defects on industrial product images in a semi-supervised training setup [77].

Chapter 3

Methods

This chapter will present the theoretical background of the anomaly detection models, feature extraction techniques and evaluation methods of interest in this research. In Section 3.1, we will give a theoretical background regarding the anomaly detection models investigated, DeepAnt and OCSVM, with an overview of the machine learning models they belong to, CNNs and SVMs. In Section 3.2, we will present the feature extraction method of interest in this research, ROCKET. Lastly, in Section 3.3, we will present the evaluation metrics and methodologies that will be used to measure the performance of the anomaly detection models.

3.1 Machine learning models

We will now go into detail about the machine learning models used in the anomaly detection task of this research, DeepAnT and OCSVM. However, to understand the two models we first need to get a fundamental understanding of the architecture groups in which they belong, CNNs and SVMs.

3.1.1 Convolutional neural network

Convolutional neural networks (CNNs) are a group of deep learning architecture that extracts high to low-level features from spatial hierarchical layers [78]. The architecture is trained through gradient descent with adapted loss function for the problem description, like common multi-layer perceptrons (MLP) [79]. What makes CNN different from the MLP is the architectural building blocks that enable them to understand spatial relationships. The architecture consists of three building blocks, *convolution*

layers, pooling layers and *fully-connected layers*. The first two mentioned are the layers responsible for building the hierarchical spatially connected features. The fully-connected layer is used to create the model output, either for regression or classification problems. The building blocks are visualized in Figure 3.1 and will be described in further detail below.

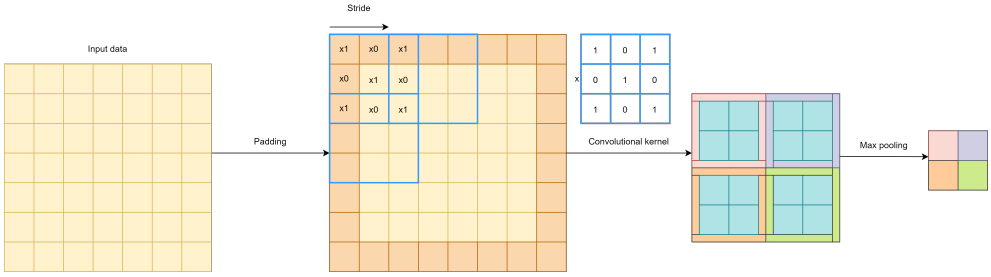


Figure 3.1: Visualization of padding, stride, convolution, and maxpooling operation.

3.1.1.1 Convolutional layer

The convolutional layers use a *kernel*, which is an array of numbers that are multiplied element-wise to the input grid of features, commonly referred to as a tensor. Summing the values of one element-wise product creates the output features for the spatial location of the output tensor. Repeating this procedure over the defined dimensions of the input tensor creates a feature map that represents different characteristics of the input tensor. Padding is often done before the convolution operation and involves adding zeros to the borders of the tensor. It enables the model to preserve dimensionality and to conduct equally many kernel operations on the borders of the tensor data as compared to the center of the tensor data, thereby reducing information loss. Before training, each convolutions layer has two hyper-parameters that have to be defined, the kernel size and the stride. The kernel size decides the number of values used by the kernel in the element-wise product. Thereby it defines the size of the input tensor that will be considered for deducing the features. The stride defines the number of steps the kernel is moved over the input tensor at each operation of convolution. During the training, the kernel values are updated through the loss value during the back-propagation of gradient descent.

It can be important to note that because CNN has been widely applied to image recognition tasks [80], the input tensor and kernel dimension are

usually defined as 2-dimensional grids. CNNs can however be applied to 1-dimensional time series as well. While 2-dimensional convolution moves the kernel over both dimensions of the input tensor during feature map extraction, 1-dimensional convolution only moves the kernel over one dimension. In univariate time series, the data is defined over the time dimension only, which makes the 1-dimensional CNNs highly applicable [81].

3.1.1.2 Pooling layer

Once the feature maps are extracted from the convolution layers, pooling is used to downsample the feature map's dimensionality and introduce translation invariance [78]. Pooling also decreases the number of parameters that have to be learned during the training of a CNN. Max-pooling is most commonly used and includes filter arrays and stride-like convolutions layers. In the max pooling operation, the maximum value of each patch in the feature map is extracted and the remaining values are removed. No trainable parameters are defined in this step, but the filter array size and stride need to be defined.

3.1.1.3 Fully connected layer

The down-sampled grid from the pooling layer is usually flattened to a 1-dimensional array and fed as input to the fully connected layer. This layer is coupled with learnable weights to the output layers, which is followed by a non-linear activation function (e.g RELU) if it's not the final layer of the network. The final fully connected layers activation function has to be adopted to the task at hand, for example, if it's a single or multi-label classification problem or a single or multi-step regression problem.

3.1.2 Support vector machine

Support vector machines (SVMs) are a group of classification and regression models that tries to find an optimal hyperplane, defined by the vector orthogonal to the hyperplane w that, for classification tasks, separates the input data into representative classes, and for regression, tries to minimize the distance of the hyperplane to the input data points. It is less common to utilize SVMs for regression tasks and we will focus on its usage for classification.
The hyperplane, or decision boundary, is found by maximizing its margin to the closest data points, or support vectors, of each class [82]. For a binary classification task as shown in Figure 3.2, we can denote the closest point

from the origin to the hyperplane as b. The dot product between w and the data points to the right side of the decision boundary will be bigger than b. Taking the dot product between w and any point to the left of the decision boundary will yield a lower value than b. Assuming that the binary classification task assigns class values $y = -1$ to the points on the right side of the decision boundary and $y = 1$ to the points on the left side of the decision boundary, and requiring a margin c, an inequality constraint for the hyperplane is described as follows

$$y_i(w \cdot x_i + b) - c \geq 0$$

Where the width of the decision boundary can be calculated as follows

$$W = \frac{x_+ - x_-}{||w||}$$

x_+ and x_- are the support vectors which are the closest points and therefore on the margin. We know that $x_i \cdot w = c - b$ and $x_j \cdot w = c + b$. Adding this to the above expression gives us

$$W = \frac{2}{||w||}$$

Since we want the hyperplane to maximize this width, the objective function needs to minimize $||w||$ subject to $y_i(w \cdot x_i + b) - c \geq 0$. This is an optimization problem that can be solved by finding where the gradient of the Lagrangian multiplier with respect to w and b is 0. It can be shown that solving the optimization problem is only dependent on the dot product of the input data x_i and x_j [83]. The resulting w and b attained by solving the optimization problem are then used for classification on new data points x as follows

$$f(x) = sign(w^T x - b)$$

Note that the above formulation presents what is called a hard margin for the SVM hyperplane and it is only applicable for linearly separable data. Many data sets are not linearly separable and even though soft margins can be used for accommodating a certain amount of errors [82], if the data is highly non-linear, then the decision boundary will likely fail in separating the classes with high accuracy. Nonetheless, many instances of non-linearly separable data can become separable in a higher dimensional space, but to avoid applying a transformation on each input data point, which can be a costly

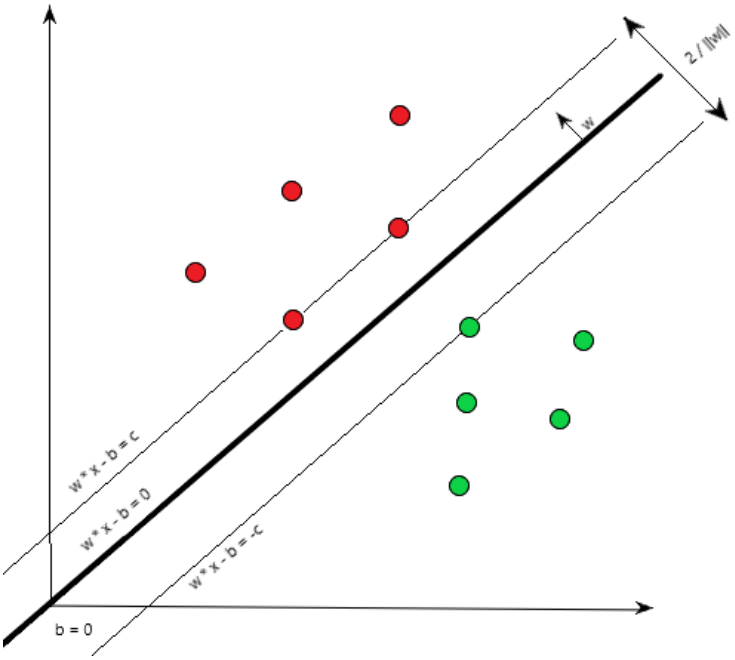


Figure 3.2: SVM visualization of features in 2D space with a 1D-hyperplane.

operation, SVMs use the kernel trick. This involves applying a transformation to pairs of input features to represent the similarity between them in a higher dimensional space [82]. Since we are only dependent on the dot product, or similarity, between the input features to maximize the margin the kernel function simply finds the similarity in a higher dimensional space without doing the transformation for each of the pairs of inputs.
There exist many choices for the kernel function, the ones that will be further investigated in the given anomaly detection task is the Gaussian radial bases kernel function

$$k(x_i, x_j) = e^{-\gamma||x_i - x_j||^2}$$

the sigmoidal kernel function

$$k(x_i, x_j) = tanh(\alpha x_i^T x_j + c)$$

and lastly, the polynomial kernel function

$$k(x_i, x_j) = (x_i^T x_j)^p$$

3.1.3 Anomaly detection models

Following the theory behind CNNs and the SVMs, we can now go into the theory behind the anomaly detection models of interest, DeepAnT, and OCSVM, and motivate why they are suited for the involved task.

3.1.3.1 DeepAnt

DeepAnt is an CNN-based semi-supervised anomaly detection model created for univariate and multivariate time series that has been shown to outperform previous density and distance-based anomaly detection models [20]. The choice of DeepAnt was based on the previous survey on multivariate time series anomaly detection methods [64]. Following current scientific problems presented in Section 1.2.2, any viable alternative needs to handle the semi-supervised anomaly detection setup following the imbalance between failure and normal production cycles and multivariate time series data. While DeepAnT fulfills these criterion, long-short-term-memory (LSTM) or LSTM and CNN auto-encoders was considered as-well because they fulfill both these criteria and have shown high performance on previous benchmark data sets. Nonetheless, the main limitations of these models involves low memory and time reliability. DeepAnt has been shown to be superior in both of these aspects and also showed similar results on several benchmark data sets [64].

The proposed model is divided into two parts, a time series predictor and an anomaly detector. The time series predictor tries to forecast time series features through regression given a preceding window of feature observations. The anomaly detector is then responsible for calculating an anomaly score for each prediction and assigning the predicted time stamps as normal or abnormal. The architecture of the two parts is described further below

3.1.3.1.1 Time series predictor architecture The time series predictor uses a CNN built for time series forecasting. To adapt the CNN to time series data, the time series is divided into history windows, consisting of continued sequences of features of dimension d, $x_1....x_{t-1}$, with label x_t in the case of single-step prediction. The value of t describes the history window size of

previous observations used to predict the forthcoming value of x_t in the time series sequence. In many-step predictions, the prediction window is instead defined as a continuous sequence of features $x_t....x_{t+p_w}$, with a length of p_w each of dimension d. The history window is fed as input to the CNN architecture and the true and predicted regression window is used to calculate the forecasting discrepancy and final loss.

For the architecture, DeepANT proposed using two 1D-convolutional layers with a kernel size of 32, ReLU activation, padding, and followed by max pooling layers. The first convolutional layer has an input size defined by the history window size t, to make it compatible with the data. The last convolutional and max pooling layers are connected to a fully connected layer responsible for the network prediction. The output size has to be adapted to the constants p_w and d. To make sure that the network output matches the shape of the prediction window size, the output layer is set to $p_w \cdot d$. The full architecture is visualized in Figure 3.3.

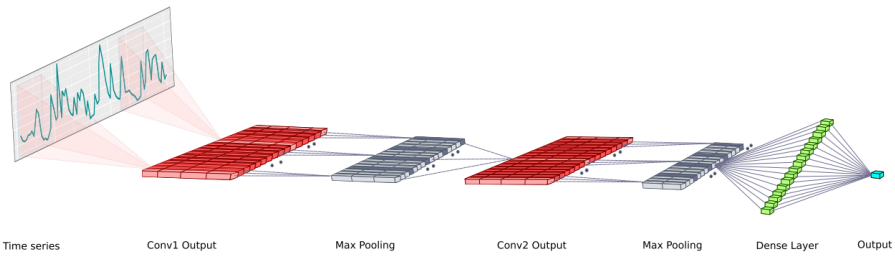


Figure 3.3: The full CNN architecture if the time series predictor in the deepANT model [20].

The architecture uses MAE for loss, as described below

$$MAE = \sum_{j=1}^{n} |x_{pred} - x_t|$$

Since the CNN architecture is implemented for forecasting, the training loss describes the discrepancy between the prediction x_{pred} and the observed true values x_t. When training on a large amount of data that mainly belongs to feature values belonging to a normal state, any large discrepancy will indicate abnormal patterns in the time series.

3.1.3.1.2 Anomaly detector During testing, the time series predictor creates a forecasting prediction $y_{pred} \in \mathbb{R}^{d \cdot p_w}$ that is compared to the true value $y_t \in \mathbb{R}^{d \cdot p_w}$. The consequent anomaly score e is calculated using the Euclidean distance as follows

$$e = \sqrt{(y - y_{pred})^2}$$

A large value for e indicates a high deviation between the expected and true observation of the time series, this is used as an indication that x_t is a possible anomaly. A threshold has to be defined for the value of e that separates normal from abnormal values, anything below the value is considered normal while anything above is considered an anomaly [20].

3.1.3.2 One-class SVM (OCSVM)

OCSVM is an extension of SVM for semi-supervised anomaly detection [84]. The choice of OCSVM was based on several previous studies using it as a baseline model for the semi-supervised anomaly detection setup [85], [86]. Thereby it fulfills the semi-supervised training setup criteria presented in Section 1.2.2 and it can handle multi-variate time series. Also, the extension of using ROCKET (see Section 3.2) for feature extraction in anomaly detection tasks has not been investigated previously, giving a reason for exploring the setup in this research.

To present the theory behind the model, OCSVM classifies data points as normal or abnormal by constructing a hyperplane in the feature space between the origin and the input data that maximizes the distance from the origin while allowing a certain amount of outliers. A 2D visualization of the hyperplane is presented in Figure 3.4. Similarly to the standard SVM, an objective function and inequality constrained are defined, but here they are defined as follows

$$f(w) = min \frac{||w||^2}{2} - p + \frac{1}{v \cdot N} \sum_{i=1}^{N} \gamma_i$$

subject to,

$$w \cdot \phi(x_i) > p - \gamma_i, \forall x_i \in X$$

Here, the learnable parameters are w and p, where w are the weights in feature space and p is a bias term for the hyperplane. γ_i is a slack variable that allows for a soft margin meaning that a certain amount of points are allowed to

lie inside the margin. v denotes the upper bound on the percentage of outliers and a lower bound for the samples used as support vectors.

As previously, finding w and b by solving the optimization problem using the Lagrangian multiplier and introducing the kernel trick, the classification of anomalous points is done as follows

$$f(x) = sgn(\sum_{i=1}^{n} \alpha_i y_i K(x, x_i) + b)$$

where a_i is the Lagrange multipliers.

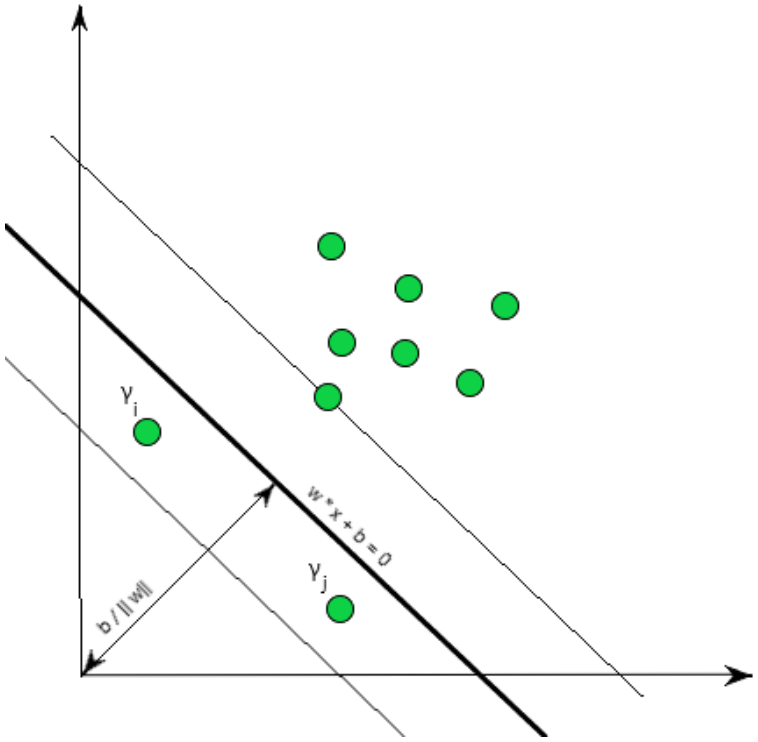


Figure 3.4: OCSVM visualization of features in 2D space with a 1D-hyperplane.

3.2 ROCKET

The ROCKET methodology will serve as a feature extraction technique for the OCSVM model. Note that if we would not find a way to address the time

dependency of the data then a simple OCSVM model would not be able to find any significant abnormality in the data outside of point anomalies. To enable OCSVM to find abnormalities in the high correlation between consecutive time points we take use of the ROCKET feature extraction.
ROCKET was introduced as a feature extraction technique for classification that tries to detect patterns associated with the involved classes in each provided time series [87]. The method was inspired by the success of CNNs, such as ResNet [88] or InceptionTime [89], and their ability to capture complex hierarchical patterns and achieve high accuracy in time series classification tasks. It uses convolutional kernels with random initialization of length (l_k), weight (w), bias (b), dilation (d), and padding that are convolved through a sliding dot product over the input time series to produce feature maps. This is done as follows

$$b + \sum_{j=0}^{l_k-1} X_{i+(j \cdot d)} \cdot w_j$$

The method can be extended from the univariate time series setting to the multivariate setting by applying the convolutional kernels to each feature in parallel.
The feature maps are then extracted by two aggregation techniques. The first one involves the maximum value and the second one is the proportion of positive values (*ppv*). Per default, 10,000 randomly initialized kernels are produced leading to 20,000 features consisting of the maximum value and ppv for each time series.

To clarify the random initialization, the original authors proposed using a kernel length sampled uniformly from the set $\{7, 9, 11\}$, weights sampled randomly from the normal distribution $\mathcal{N}(0, 1)$, bias sampled randomly from the uniform distribution $U(-1, 1)$, and a dilation of $d = \lfloor 2^x \rfloor$ where x is sampled from $U(0, A)$ ($A = log_2 \frac{l_i - 1}{l_k - 1}$). This ensures that the kernel length with the dilation is always smaller than the input time series. The presence of padding is chosen randomly, and if chosen, zeros are padded to both ends of the time series to allow the middle element of the kernel to be centered on each point of the time series. Indeed, without padding, the kernel can not be centered at the beginning and end of the time series causing a greater focus towards the center.

By random initialization of the kernel parameters, the different kernels can

capture different characteristics of the time series. For instance, dilation determines the spread of the kernel through insertion of empty values based on the randomly sampled frequency. A dilation of three would therefore mean that every third second of the input time series would be convolved with the kernel weights. Consequently, a large dilation captures low-frequency patterns while a small dilation captures high-frequency patterns.
Each feature map produced by a randomly initialized kernel thereby represents the extent to which the pattern represented by the kernel is present in the time series. For large output values of the kernel operation, a higher presence of the pattern connected to the kernel exists.

The convolutions have been shown to capture similar patterns as other time series feature extraction methods, such as shapelets [90], but require a fraction of the computational time [87]. This further motivates the usage of ROCKET in this research following previous work that have combined the usage of OCSVM with shapelet transformations [91]. Since ROCKET has been shown to find similar patterns to shapelet transformations, but with a fraction of the training time, it could serve as a more efficient feature extraction methodology.

3.3 Evaluation methods and metrics

Following model training, the evaluation step includes the usage of evaluation metrics for testing the generalized performance of the given model, comparing the results of unseen and seen samples. The choice of evaluation metrics is a key concept, especially in the given task considering the usual imbalance between failure and functional states of machine data in manufacturing settings [92]. The imbalance of the data set needs to be considered in the choice of metrics and following previous recommendations [93], the metrics that will be presented, Confusion matrix, Precision, Recall, and f1-score, are highly relevant for the involved task. Furthermore, statistical tests such as the Wilcoxon signed rank test have also been a popular evaluation method used for anomaly detection [94]. Lastly, in high multivariate settings, the nature of anomalies can be hard to interpret and a method previously used in visualizing high dimensional data to reason about the characteristics of anomalies is t-SNE [95].

3.3.1 Confusion matrix

A confusion matrix is an evaluation metric used in classification tasks that can be used in both binary and multi-class classification tasks. It is constructed by a NxN matrix where N is the number of class labels in the data. A demonstration of a confusion matrix for a binary classification problem, with a negative and positive class, is presented in Figure 3.5. Note that the columns represent the true labels while the rows represent the predicted value. True positive (TP) represents the number of correctly classified positive instances, false positives (FP) represent the number of samples that are classified as positive while belonging to the negative class, false negatives (FN) represent the number of samples that are classified as negative while belonging to the positive class, and true negative (TN) represents the number of correctly classified negative instances.

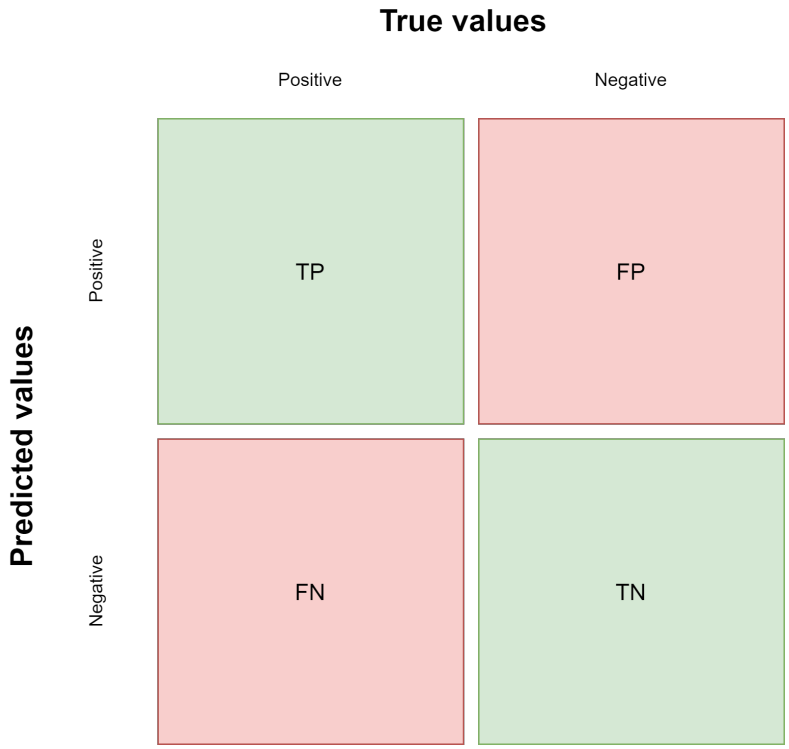


Figure 3.5: Visualization of the confusion matrix in a binary classification task (negative and positive class).

This introduction of TP, FP, FN, and TN further leads to the continued metrics precision, recall and F1-score.

3.3.2 Recall, precision and F1-score

Recall and precision are metrics commonly used in time series classification and anomaly detection settings [96] due to their ability to consider the imbalance of data sets. For binary classes, Precision is defined as follows

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

and Recall is defined as follows

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

Precision thereby defines the fraction of all detected positive instances that are real positives. Recall defines the fraction of real positive instances that is detected. These definitions can also be extended for negative instances. Precision is then defined as follows

$$Precision = \frac{TN}{TN + FN}$$

and Recall is defined as follows

$$Recall = \frac{TN}{TN + FP}$$

The F1-score is then a measure of the overall accuracy based on precision and recall. It is defined as follows

$$F1 - score = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

3.3.3 Wilcoxon Signed-Rank Test

Wilcoxon Signed-Rank Test is a hypothesis test that can be used for pairwise comparison of independent samples. When comparing independent samples the hypothesis test tries to answer if there is a statistically significant difference between the median of the samples [97]. It does so by introducing the null hypothesis that there is no difference between the two sample means. To test this hypothesis, the sum of the signed ranks is used for test statistics

$$W = \sum_{i=1}^{N} sgn(x_{2i} - x_{1i}) \cdot R_i$$

Here, x_{1i}, x_{2i} denotes the measurement i in the first and second samples. R_i denotes the rank of the pair x_{1i}, x_{2i}, particularly, the position of the pair based on the ordered list $|x_{1i} - x_{2i}|$. Lastly, to test the null hypothesis, the critical value based on the sample size N and significance level α is compared to the value of W. If it's smaller, then the null hypothesis is rejected, which means that there is a statistically significant difference between the median of the two samples.

3.3.4 t-SNE

t-SNE is a dimensionality reduction technique that maps high-dimensional data into a two or three-dimensional grid space while retaining the characteristics of the data. If we consider an instance $x_i \in \mathbb{R}^d$ belonging to data set x, the first step of t-SNE is the calculation of the Euclidean distance of x_i to remaining instances in x. The Euclidean distances are then mapped to a joint probability space where the probability $P_{i|j}$ represents the probability that the instance x_j would be picked as a neighbor to x_i, if the process of picking a neighbor for x_i followed a Gaussian distribution centered at x_i [98]. Calculating the probability $P_{i|j}$ is thereby done as follows

$$P_{i|j} = \frac{e^{(-|x_i - x_j|)^2/2\sigma_i^2}}{\sum_{k \neq l} e^{-|x_l - x_k|)^2/2\sigma_i^2}}$$

Where σ_i is the variance of the Gaussian function centered at x_i. The σ value has to be adopted for every data point in the time series. Indeed, in denser regions, a smaller σ value should be prioritized compared to sparser regions. Consequently, the σ value is calculated through a binary search with the perplexity provided for the specific task.

After the Gaussian distributions have been defined for each instance in the data, they will be mapped to a lower dimensional space. This is done by adapting the student t-distributions to a similar probability as the Gaussian distributions but in a lower dimensional space

$$Q_{i|j} = \frac{(1 + (|x_i - x_j|)^2)^{-1}}{\sum_{k \neq l} (1 + (|x_l - x_k|)^2)^{-1}}$$

To optimize this distribution the Kullback-Leibler divergence [99] is used. The high-dimensional instances are then placed in the lower-dimensional space according to the probability distribution Q.

Chapter 4

Implementation

This chapter will present how the proposed anomaly detection methods was adapted for the task. An overview of the methodology is presented in Figure 4.1 and will be used as a foundation in the methodology description. In Section 4.1, the dataset, the splitting of the data set, and the initial data cleaning and processing common for both models is is presented. It also present the implementation of the ROCKET feature extraction used only for the OCSVM. In Section 4.2, the full setup for the DeepAnT and ROCKET OCSVM will be presented, including training methodology, model adaption, and hyperparameters tested. In Section 4.3, the evaluation process for the two models is described. Lastly, in Section, 4.4 the setup of the experiential environment is presented for reproducibility purposes.

4.1 Data processing

The data processing method involves gathering the time series data and processing it to a suitable format for the different models. Before going into the processing step, I will introduce the structure of the dataset investigated.

4.1.1 Data-set

Time series data from a CNC-machine referred to as machine 50 has been collected by the company Nytt following the MTConnect protocol for a duration of 6 months, starting from August 2022 to January 2023, with a frequency of 1 second. The MTConnect protocol is a technical standardized

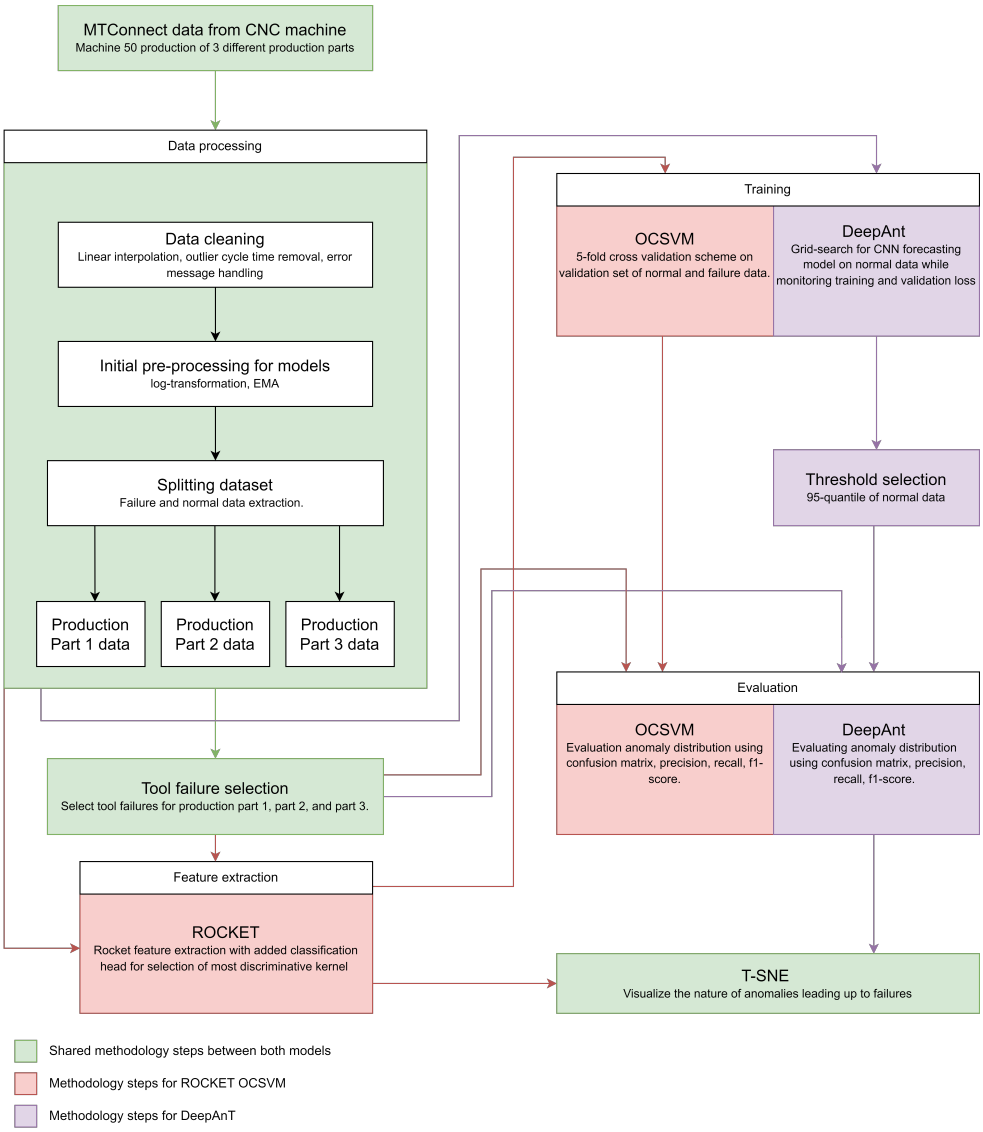


Figure 4.1: Overview of methodology separated into three parts *Data processing, Feature extraction, Model training* and *Model evaluation*

data retrieval protocol and supports extraction of multiple analytical features connected to machine working status [13]. An overview of the full set of features is presented in Table 4.1. Note that the highlighted rows are the ones of importance to the given task, below is a further description of some of the attributes to understand how failures can be deduced from the data.

- **state**: This is a categorical attribute taking the values *ACTIVE,*

STOPPED and *INTERRUPTED*. Normal working states will be labeled *ACTIVE* which indicates that the machine is running without any observed failures. The *STOPPED* label means that the machine is inactive with the specified reason being presented in *EmergencyStop*.

- **EmergencyStop**: As mentioned, this column present the current stoppage if *state* is *STOPPED* and it is not a short stoppage. If *state* is *ACTIVE* the last occurring failure will instead be presented. The emergency stops are divided into four comprehensive categories, *operator, tool, machine* and *other* stoppages. In this research, we will focus on the tool stoppages. These stoppages are related to issues in the current tool that is activated and specified by the *line* column during the production of the current part. Operator stoppages are simply stoppages initiated by the operators and are thereby not predictable through data monitoring.

Lastly, through the available domain knowledge of operators, *Xload, Yload, Zload* and *RotatoryLoad* have been concluded to be the most reliable features to indicate the health status of the production machine.

To give a better overview of the time series structure refer to Figure 4.2. As can be seen, the time series can be divided into both the part being produced, production cycles, as well as the tools being active in the production process. A production cycle constitutes the process of finishing one part and the tools are the involved instruments used in the process, including drillers, cutters, grinding, and turning tools.
Different parts being produced can follow different cyclic patterns over the production cycles, it is therefore important to group the data into the different parts being produced and train separate machine learning models for each part, since the generalization that otherwise would be needed to model the normal state data of each production part can affect the anomaly predictions negatively. To enable this, a new column was created with a unique index for each production cycle as well as information on which production part each of the production cycle belongs to. Furthermore, columns holding the normalized time of the production cycle, categorical data regarding the active production cycle tool, and the normalized time the tool has been active, were added. An overview of the processed features used as input to the different models is presented in Table 4.2. The motivation behind adding these features

Features	Description
mtConnectId	Identifier of the procol verision.
machineId	Identifier id of the machine.
State	Current state of the machine.
Partcount	Binary value indicating that the current production cycle is complete.
ControllerMode	Indicates if the controller is off or on.
RotatoryLoad	Pressure to the cutter from the object being feed to the machine.
PathFeedrate	The cutter speed being engaged onto the production part (units/min).
RotatoryActVelo	Rotational speed of cutter
RotatoryOverrideVelo	Binary value indicating if rotatory active velocity was changed by operator
Xload	Feeding load on the machine on the x-axis.
Yload	Feeding load on the machine on the z-axis.
Zload	Feeding load on the machine on the z-axis.
originalTime	Time-stamp consisting of yaer,month,day,hour,minute and second for the current measurement.
EmergencyStop	Description of the last occured, non-short, machine stoppage.
PathPosition	Position with x,y, and z coordinates on the surface of the part.
line	Tool name being activated or program part that is currently being executed.

Table 4.1: MTConnect time series features

was based on the cyclic behavior over production cycles. As can be seen in Figure 4.2, tool changes can be followed by a large change in feature values. The introduction of dummy variables has previously been shown to increase the predictive performance of volatile time series due to cyclic behavior [100]. Lastly, the tool errors that will be investigated are chosen based firstly on the operator domain knowledge. Some tools have shown more issues in the production of specified parts than others and are therefore of greater interest to investigate. This involves tool *T606*. Secondly, the tools were chosen based on the availability of previous failure data and the reason for breakage. Several tool errors include *tool life-count end* alarms, which simply indicate that the

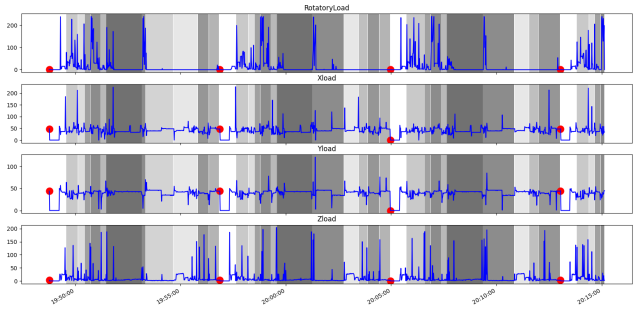
tool has reached its recommended number of production cycles. In several cases, this is not visible through the data features, and therefore these tool failures were excluded. In the end, the remaining tools chosen for investigation were *T505* for production part 1, *T161* and *T404* for production part 2, and *T1228* and *T199* for production part 3.

Features	**Description**	**Input to**
originalTime (index)	Time-stamp consisting of year,month,day,hour,minute and second for the current measurement.	DeepAnt, ROCKET
RotatoryLoad	Pressure to the cutter from the object being feed to the machine.	DeepAnt, ROCKET
Xload	Feeding load on the machine on the x-axis.	DeepAnt, ROCKET
Yload	Feeding load on the machine on the z-axis.	DeepAnt, ROCKET
Zload	Feeding load on the machine on the z-axis.	DeepAnt, ROCKET
line-TXXX	Binary value representing if tool *TXXX* is active.	DeepAnt
...	...	DeepAnt
line-TYYY	Binary value representing if tool *TYYY* is active.	DeepAnt
cycle-proc	Normalized time in seconds the production cycle has been active.	DeepAnt, ROCKET

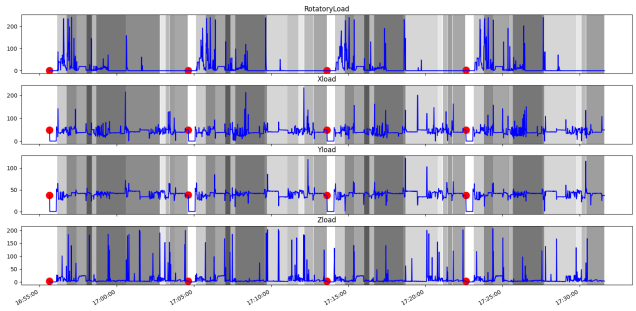
Table 4.2: Extracted features used for training. Note that row 6 represents the other tools activation status. The number of tools might different dependent on production part.

4.1.2 Data cleaning

After gathering the manufacturing data, data cleaning was of the essence to make it compatible with the anomaly detection models. One major challenge present in both this data and many other real-world time series data sets is missing values [101]. To handle missing time steps in the data, linear interpolation was implemented. The choice was motivated by its common



(a) Part 1 production



(b) Part 2 production

Figure 4.2: RotatoryLoad, Yload and Zload for four production cycles in the machine 50 during production of two parts. The red dots show the end of one production cycle and the background highlights show the difference tools being active in the production cycles.

usage in sensory data handling [102]. The method involves replacing missing values with the intersection point of the timestamp and the line interpolated between the first available point previous to, and post, the missing timestamp as follows

$$f_{t,i} = \frac{f_{t-1,i}((t+1)-t) + f_{t+1,i}(t-(t-1))}{(t+1)-(t-1)}$$

Here, $t+1$ is the next time-stamp of the time series, $t-1$ is the previous time-stamp of the time series, $f_{t+1,i}$ is the features i value at time-stamp $t+1$, and $f_{t-1,i}$ is the features i value at time-stamp $t-1$.

Next, the data was cleaned from downtime connected to operator stoppages. This was motivated by the fact that these stoppages are not connected to any characteristics that could be predicted through the data, as well as causing high amounts of downtime that the anomaly detection model would be trained on. Further anomalous cycles characterized by exceptionally long production times were also removed. Each production cycle connected to the different production parts has a set time that it is expected to run, however, cumulative short stoppages can also cause large downtime. Since the models that will be investigated follows a semi-supervised learning scheme where we only train on normal production cycles, we want to clean the data out of any abnormal patterns present in the normal data that can not be linked to failures. Therefore any cycle with length L, belonging to production part p, that does not follow the criteria below was removed

$$L_{p,\sigma} - L_{p,\mu} \leq L \leq L_{p,\sigma} + L_{p,\mu}$$

Here, $L_{p,\mu}$ is the mean length of the production cycles for a given part p, and Lp, σ is the standard deviation of the production cycles for a given part p. Note that 0.02 % of the top and bottom production cycle length was clipped before calculating the limits to handle large outlier cycle times.

Lastly, incorrect error messages were handled. As described in Table 4.1, the EmergencyStop column of the MTConnect protocol presents either the last occurred stoppage, or the current stoppage, if it is not a short stoppage. Short unlabeled stoppages occur in many production cycles and to make sure that these are not labeled as failure production cycles, the EmergencyStop label was changed for all stoppages with a time frame of less than ten seconds. This allows failures to be extracted with safety based on only the EmergencyStop and State features.

4.1.3 Initial pre-processing

After the data cleaning, a common pre-processing scheme was implemented for both of the models. This is a standard procedure when handling time series data and common techniques used including power transformation to handle skewness, differencing for removal of trends, and normalization or standardization of input features [103]. Before conducting any pre-processing, exploratory data analysis of the cleaned time series was done to get further insights into its properties. Observing Figure 4.2, it can be concluded that the raw data features are volatile and, as can be seen observing the feature value

distribution in Figure 4.3a, the data distribution was highly skewed showing a clear zero-inflated distribution for several features [104]. These factors are caused by the cyclic sudden increases in feature value over the production cycles. These could be falsely labeled as anomalous since they are infrequent globally over the time series, but normal considering that they repeat over most production cycles. Furthermore, normalization will become affected by the big-scale difference caused by these sudden increases.
Because of these factors, a combination of feature-wise log transformation and exponential smoothing was applied as a pre-processing step. As mentioned previously, log transformation has been shown to handle data skewness and it also shortens the sudden peaks in feature values that should not be considered anomalous. Exponential smoothing is applied to handle noise and uncover the underlying trend of the time series. Indeed, previous studies have discussed how it can be an important pre-processing step for handling random fluctuation and reveal the true inclination of the series previous to anomaly detection [105]. The two steps to calculate the new value $\hat{x}_{i,t_p+1|t_p}$ can be summarized as follows

$$x_{i,t} = log(y_{i,t} + c) + c$$

$$\hat{x}_{i,t_p+1|t_p} = \alpha x_{i,t_p} + \alpha(1-\alpha)x_{i,t_p-1} + \alpha(1-\alpha)^2 x_{i,t_p-2}...$$

Where, $y_{i,t}$ is the original observation for feature i at time t, c is a constant factor (chosen as 1 in this research) used to handle 0 or negative valued features for the log-transformation, t_p is the current timestamp in the given production cycle that is being processed, and α is a weight term deciding the degree to which we consider previous observations. A α-value closer to 1 gives higher weights to more recent observations, which is beneficial if we want to keep certain volatility, while a value closer to 0 gives more weights to previous values and is therefore more suited for capturing underlying trends. Since we want to prioritize finding the trend of the time-series, a value closer to 0, 0.3, was chosen in this research.

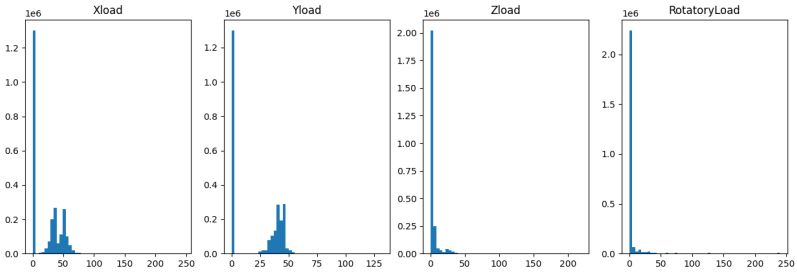
Lastly, since we want each feature to give equal weights in the training, we apply min-max normalization as follows

$$x_{i,t} = \frac{x_{i,t} - x_{i,min}}{x_{i,max} - x_{i,min}}$$

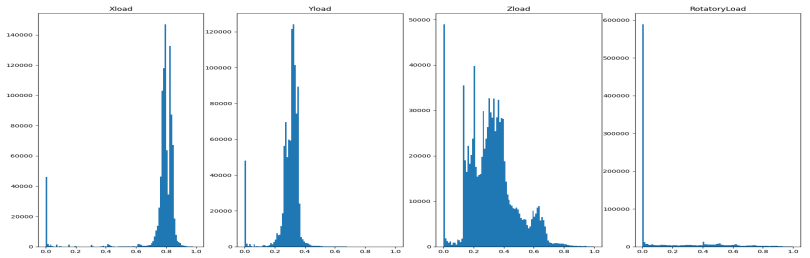
Where i is the feature number, $x_{i,min}$ is the minimum value for the feature i, i, $x_{i,max}$ is the maximum value for the feature i, and $x_{i,t}$ is the feature value

at the current time step that is scaled. The resulting distribution of the features is presented in 4.3b.

Note that the pre-processing can further be motivated through the characteristics of anomalies. While smoothing the time series can cause some point anomalies to be less apparent, this did not cause large issues since the normal working condition seemed to have plenty of point anomalies not connected to failures. Any transformation that enables us to find the trend of the normal production cycles and separate them from production cycles close to failures is beneficial for the anomaly detection task. It can be worth stating that an alternative approach that was considered was to apply time series decomposition. However, the production cycles do not follow a precise seasonal pattern, thereby this method was not feasible. Also, there is a possible information loss connected to the pre-processing used. Ideally, we want to only remove noise and volatility that is not connected to any anomalies in the time series, however, it is not guaranteed.



(a) Data distribution previous to pre-processing



(b) Data distribution after pre-processing

Figure 4.3: Feature distribution over the involved features Xload, Yload, Zload, and RotatoryLoad for production of part 1.

4.1.4 Splitting dataset

Perhaps the most important part to understand the methodology is the separation of the time series into the pre-failure anomalous state and the normal working condition states. Following the research questions, we wanted to investigate the anomaly distribution in production cycles that leads to failure and in production cycles that do not lead to failure. We also wanted to investigate how far-back anomalies connected to failures can be uncovered, and how many production cycles that needs to be removed pre-failure to ensure that we do not train the semi-supervised models on anomalies connected to failures.
To separate the data into the pre-failure anomalous states and normal-working conditions, for each of the three production parts, the data sets were divided into normal working condition data (N) and anomalous data close to failures (F). N is defined by the number of production cycles removed before every failure n_w, which is the parameter that enables us to investigate the impact of training on different time windows in connection to failures. F is defined by the number of production cycles that should be labeled as anomalous f_w. This parameter was introduced to investigate in what time frame possible anomalies occur that could indicate a future failure. F is also further divided into the different tool failures since, as presented in Section 4.1.1, we will investigate different tool failures individually. The division of the time series for each production part is presented in Figure 4.4. The values investigated for each parameter are presented in Table 4.3. Note that f_w was chosen as eight and the value was motivated by the expected production cycle length. The production cycles are approximately seven minutes long and if anomalies can be uncovered in the eight production cycles before failure, this would give sufficient time for reactive maintenance on the machines.

Hyper-parameter	**Values**
Pre-failure data removed for N	$9, 100, 200$
Pre-failure data belonging to F	8

Table 4.3: Hyperparameters for the semi-supervised setup of each model investigated.

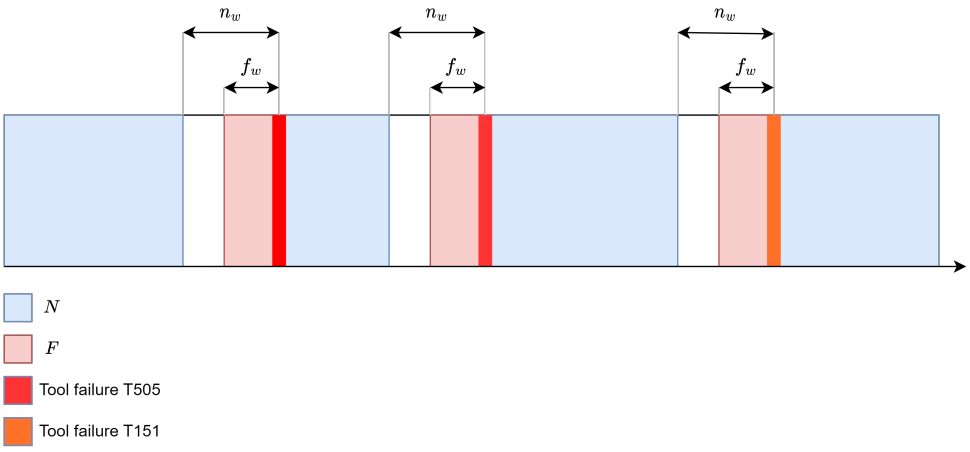


Figure 4.4: Visual representation of the division of the original time series into the normal working production cycles N, and the pre-failure production cycles F. Note that the same division is repeated for each production part time series.

4.1.5 ROCKET

The proposal of ROCKET was, as stated in section 3.2, originally proposed for time series classification. This anomaly detection problem is however treated as a classification task where we label defined time windows that are not in close proximity to failures as non-anomolous and production cycles in close proximity to failures as anomalies. The pre-processing then involves training the ROCKET-feature extractor with time series sequences $s_i = x_i, x_{i+1}...x_{l+i}$, where l is the investigated time window of the time series, from the normal working state data.

As can be seen in Table 4.2, the categorical features used for the DeepAnT model is excluded from the training, since the use of categorical features are not discussed in the original proposal of ROCKET. Furthermore, because of the high dimensionality of features extracted from ROCKET, a classification layer was added on top of the kernels that was trained to predict the class belonging of each time sequence. The top 2 % of the convolutional kernels with highest weights for the classification task was then selected as features for the anomaly detection method. This is a filtering feature selection approach and it is important to note that the hold-out set of 20 % of F belonging to each tool failure for the specified production part was used for this process together

with the training data for the OCSVM (see section 4.2.2 and Figure 4.6).

The amount of time-steps l involved in the sequences are a hyper-parameter that needs to be investigated. When choosing this hyper-parameter, we need to consider that production cycles can have varying length and that failures can lead to early stoppages. Since there might be valuable information close to failures that we want to investigate, the sequence size was chosen as half and a third of the length of the production cycles leading to failures.

4.2 Adapting models to data

In the following section, we will present how the investigated models DeepAnt and OCSVM was adapted to the data.

4.2.1 DeepAnt

Firstly, to test the generalized performance of the DeepAnT model, we need to divide the dataset into the training set, validation set, and testing set. The datasets connected to each part were thereby divided into 80 % of data from N used for training. This set was further divided into 90 % for model training and 10 % for validation monitoring. The remaining 20 % from N was used for testing and all the data sequences from F, belonging to each specific tool failure, were used for testing. This follows the semi-supervised approach where we want the model to learn the normal working condition of the machine and test it on the anomalous production cycles connected to failures. Note that each data set was split based on production cycles to keep the time dependency. Next, a sliding window approach was used on the production cycles of the training, validation and testing sets. This is a commonly used technique for transforming time series into suitable structures for machine learning models [106] and involves iterating over the time series with a stride of 1 and dividing it into separate inputs based on an assigned history window size, and label, based on the predictive window size.

For the model setup, the original architecture was replicated, using two convolutional layers with 32 filter kernels, max-pooling, and Relu-activation. For training, Adam-optimizer was used and the learning rate was fixed at $0.5 \cdot 10^{-4}$. MAE was used as the loss function and the model training was conducted over 10 epochs with batch size 32, which was chosen through investigation of the convergence in the loss plots in pilot trials. Only two

updates were done to the original architecture, the first one in regards to the anomaly detector. The anomaly threshold was set automatically, and not through trial observations, using the 95-quantile rule. This method was motivated by the fact that it would be time-consuming to manually assign thresholds to each hyperparameter setup of the models. A visual demonstration of the approach is shown in Figure 4.5. Secondly, following Figure 4.3, the feature RotatoryLoad is still imbalanced after the pre-processing. This is caused by several intervals of each production cycle keeping a constant value of 0. These regions are of less interest to the anomaly detection method since it simply indicates that the specific type of load is not present currently during the production process. This can not be further handled through pre-processing of the data but could cause the forecasting model to become over-adapted to the dominating constant regions for the attribute. Therefore, we introduced the method of DenseWeights [107], which weights the loss function based on the mean probability of observing the target values of each feature based on kernel density estimations. This is a novel method for zero-inflated distributions that prioritize rare samples, which, for RotatoryLoad, consist of the active regions of the production cycles.

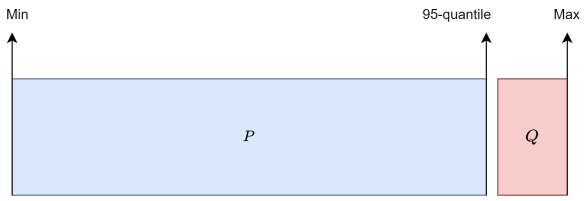


Figure 4.5: Visualization of the automatic anomaly threshold assignment in the training data. Q is the anomaly scores belonging to the predictions that will be labeled as anomalies while P is the anomaly scores belonging to predictions that will be labeled as normal. The anomaly score with the 95-quantile value in the training set is used as threshold and is extended to the testing set.

The adaptable parameters for DeepAnT involve both the history window, meaning the number of observations to consider for the prediction, and the predictive window, meaning the number of future observations that should be predicted by the model. An overview of the parameters investigated in the grid search for the DeepAnt model is presented in Table 4.4. The history window was chosen based on the memory availability of the experimental setup.

The predictive window was chosen as one and ten to investigate the anomaly detection capabilities dependent on single and multi-step predictions.

Hyper-parameter	**Values**
History window	$30, 60$
Predictive window	$1, 10$
n_w	$9, 100, 200$

Table 4.4: Hyper-parameters investigated for the DeepAnT model

4.2.2 OCSVM

To reach the optimal performance of the model, the parameter ν, deciding the upper bound on the percentage of outliers and a lower bound for the samples used as support vectors, and the kernel function used for creating the decision boundary needs to be decided. The ν was selected to be 0.01 to set a strict limit on the number of false positives. An overview of the parameters investigated in the grid search for the OCSVM model is presented in Table 4.5. Note that since we are using a one-class classifier, a new grid-search scheme is needed. As explained in section 3.1.3.2, the OCSVM is trained on only normal state data, which will cause several issues for ordinary grid-search models [108]. Using metrics such as precision is not possible when only one class is present, also, other metrics such as accuracy can be misleading. The accuracy might be higher for a kernel function that sets a less restrictive limit on the hyperplane adapted to the normal working state data, which could result in a small number of true positives, meaning anomalies labeled as anomalies. Therefore, we evaluate the model with a new k-fold-cross-validation scheme, where 80 % of the normal working state data, and a hold-out set constituting 20 % of the failure data is used to calculate the recall score of each k-fold model based on both the normal validation data N_{val} of each fold, and the failure hold out set F_{val}. The kernel function is then chosen based on the recall over both classes. A visualization of the scheme is shown in Figure 4.6. The final model training is then done with the kernel function chosen from the cross-validation scheme, using the same 80 % of the normal training data, then evaluated the model with the remaining 80 % of the failure.

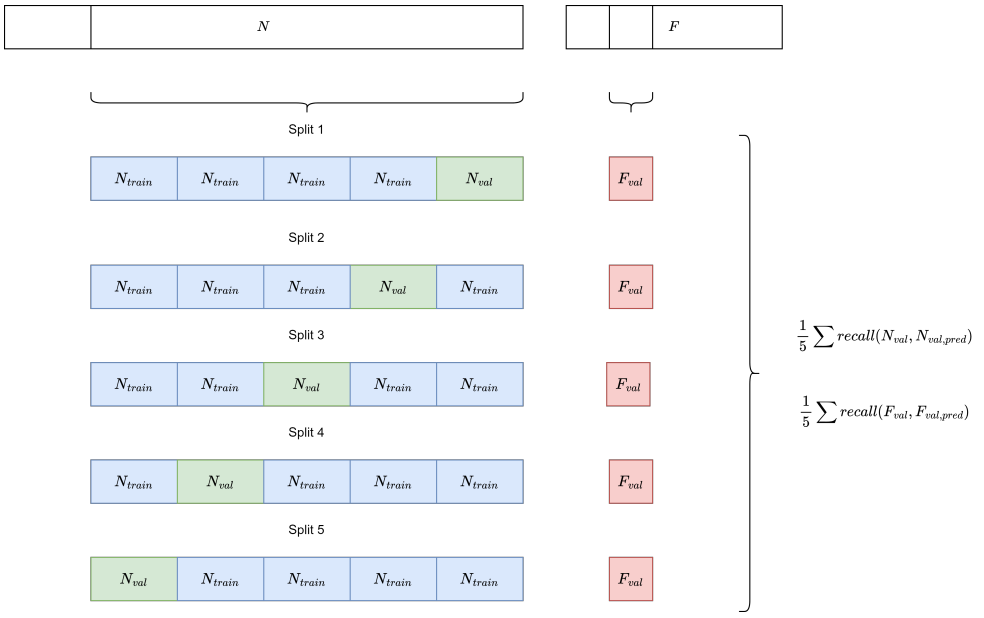


Figure 4.6: Visualization of the k-fold cross validation scheme implemented for the OCSVM training. Here, N is the normal working condition data, F is the pre-failure data, N_{val} is the validation data used for the recall calculation, N_{train} is the part of N used for training in each k-fold model, F_{val} is the validation failure data used for the recall, $N_{train,pred}$ is the predicted values of N_{val}, and $F_{val,pred}$ is the predicted values of N_{train}.

Hyper-parameter	**Values**
Kernel functions	rbf, sigmoid, poly
ν	0.01
n_w	$9, 100, 200$

Table 4.5: Hyper-parameters investigated for the OCSVM model

4.3 Evaluation

For the evaluation of the DeepAnt model, training and validation MAE loss will be monitored during training to investigate which parameter setup that provides the highest performance in terms of forecasting ability. Another set of trials is conducted with each parameter setup where we measure the classification accuracy of normal working conditions and abnormal working conditions pre-failure using both confusion matrix, f1-score, recall, and precision. The metrics are motivated by the high-class imbalance, which

needs to be taken into account through the evaluation. Note that other metrics and methods, including ROC-AUC [109] or comparing the anomaly scores of the pre-failure production cycles and the normal production cycles using a statistical test such as the Wilcoxon signed ranked test, were considered but excluded from the evaluation. AUC-ROC depends on the classification probability of the involved classifier and knowing that OCSVM and DeepAnt are both semi-supervised models trained on one class, we don't have access to this. The Wilcoxon signed ranked test was implemented and tested but excluded considering the problem of over-powering in statistical tests [110]. Indeed, with the large number of predictions of anomaly scores done between pre-failure and the normal working state of the machine, the sample size will be large enough to detect any small deviations as statistically significant when comparing the two populations. Furthermore, since OCSVM do not have access to an anomaly score, we would only be able to realize this for the DeepAnt model.

To draw further conclusions regarding the pre-failure production cycles, the forecasting predictions of DeepAnT will also be used to train a t-SNE model. This is done since the characteristics of the data close to failures is unknown. They could constitute an abundance of anomalies or a small proportion of anomalies that are of great importance to alarm for possible future failures. However, the second case is only useful for manufacturing companies if they can be separated from possible anomalies in the normal working state data. Using t-SNE, we can visualize the distribution of prediction errors on a lower dimensional space and thereby observe potential discrepancies between pre-failure and normal production cycles. This also allows us to reason about how early anomalous segments appear, given that we can include labels in the visualization for each of the production cycles leading up to failure.

ROCKET OCSVM will be tested through the same metrics, and the predictions will be visualized through another t-SNE model. However, for the ROCKET OCSVM, we will train the t-SNE on the ROCKET features.

It is important to understand that the results will be based on experimental investigations of the production cycles previous to failure. Little to nothing is known about the anomalies in production cycles leading up to failures or in the production cycles that are not connected to failures. The labeling of pre-failure time steps as anomalous and consequent use of classification metrics are merely done to investigate to which extent anomalies can be used

to indicate machine failure based on the distribution of anomalies in normal and pre-failure production cycles.

4.4 Experimental Setup

The entire set of experiments was conducted on a Windows 64-bit operating system machine with 16 GB RAM and A NVIDIA GeForce GTX 1060 6GB GPU enabled for model training. The pre-processing, feature extraction, model implementation, and evaluation matrix were each implemented in Python version 3.9.1 with the list of libraries used in the virtual environment for the project presented in Table 4.6.

Library	**Version**
Numpy	1.23.5
pandas	1.5.3
scipy	1.10.0
seaborn	0.12.2
torch	1.13.1+cu116
statsmodels	0.13.5
lightning-utilities	0.6.0.post0
scikit-learn	1.2.1
matplotlib	3.6.3
sktime	0.16.1

Table 4.6: Summary of the libraries used in the virtual environment where the implementation was done.

The pre-processing and data collection methodology was enabled through a combination of numpy, pandas, and sklearn. Pytorch (pytorch-lightning) was used for the DeepAnt model implementation while the OCSVM, PCA, t-SNE, and evaluation metrics, were implemented through sklearn. Lastly, the ROCKET feature extraction was implemented through sktime.

Chapter 5

Results and Analysis

In this chapter, we present the results following the methodology presented in section 3. We will begin with presenting the results for the DeepAnT in Section 5.1. We will then continue to the ROCKET OCSVM In Section 5.2.
It is important to re-iterate the separation of the data following the methodology presented in Section 4.1.4, to understand what testing data that is labeled as anomalies and what data is labeled as non-anomalies. Throughout the results, we will refer to the pre-failure labeled anomalous production cycles as F, and the normal working condition production cycles as N. Furthermore, the DeepAnt evaluation firstly involves investigating the forecasting performance of the CNN-architecture to conclude how well it learns to model the MTConnect time series data. However, no model selection is done in this step since the training and validation performance is only dependent on N and we cant draw conclusion regarding the anomaly distribution of F following the training loss. Thereby we will present the anomaly predictions of the pre-failure and normal working condition production cycles for every parameter setup to conclude if there exist a difference in the distribution.
For the ROCKET OCSVM, model selection is done during the 5-fold-cross validation since it includes data from both N and F and thereby possible separation between the two subsets. After the model selection, the distribution of anomalies in N and F will be investigated.

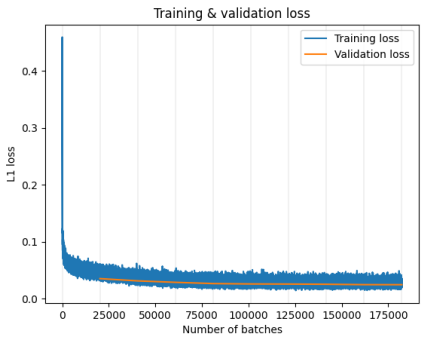
5.1 DeepAnt

In this section, we will begin with presenting the results related to the training of the CNN forecasting model of DeepAnT dependent on the parameter setup.

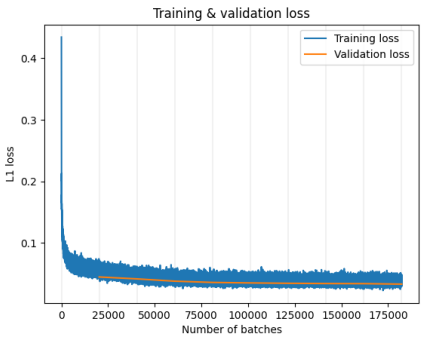
We will then continue to investigate the chosen models anomaly detection results for N and F.

5.1.1 Forecasting results

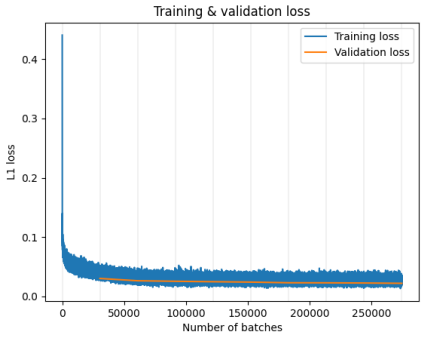
For the DeepAnt model, following the methodology presented in Chapter 3, we began with training the CNN forecasting model on the normal working condition data and monitoring the training and validation loss. A subset of the models monitored training and validation loss are shown in Figure 5.1. The full set of trial results, including final validation loss, training loss, and anomaly threshold values, are presented in Table 5.1.



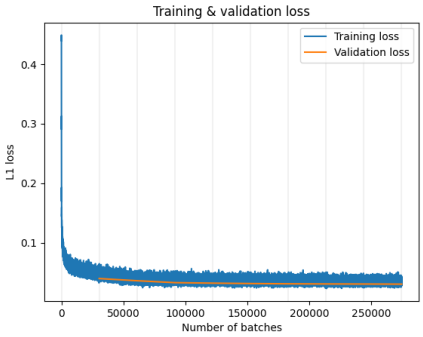
(a) **Part 1:** h_w: 30, p_w: 1, n_w: 9



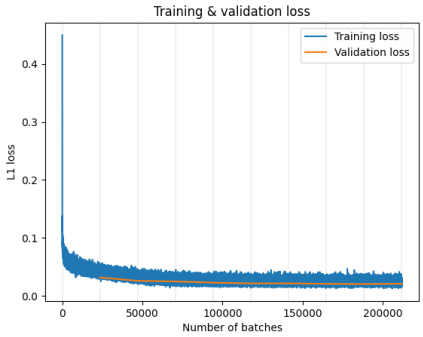
(b) **Part 1:** h_w: 30, p_w: 10, n_w: 9



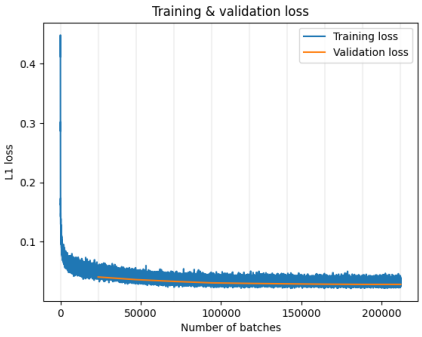
(c) **Part 2:** h_w: 30, p_w: 10, n_w: 9



(d) **Part 2:** h_w: 30, p_w: 10, n_w: 9



(e) **Part 3:** h_w: 30, p_w: 10, n_w: 9



(f) **Part 3:** h_w: 30, p_w: 10, n_w: 9

Figure 5.1: Training and validation loss for models with the specified hyperparameter setup. Note that L1 loss is synonymous with MAE and h_w refers to history window.

From these results, firstly, we can conclude that independently of the production part the history windows have little to no effect on the final loss and anomaly threshold. Interestingly, the smaller history window of 30 results in a smaller training loss, validation loss, and anomaly threshold, in the majority of the cases. This indicates that a smaller history window models the normal production cycles with higher accuracy compared to the larger predictive window of sixty. Secondly, the generalized performance comparing the training and validation loss is very high. Indeed, there is little to no difference between the validation and training loss, indicating that the model performs well in modeling normal working conditions not seen during model training. Thirdly, the forecasting performance is very similar over the different production parts.

For the number of pre-failure cycles excluded the effect varied for production parts. The best results for production parts 1 and 3 were reached with 9 production cycles excluded. For part 2, the anomaly threshold and validation loss were similar with 9 and 100 production cycles removed. In general, the investigated numbers of production cycles pre-failure excluded from training seem to have little to no effect on the predictive performance, meaning that the same loss and anomaly threshold is reached independent of the parameter value.
The predictive window however has a major effect on the training. The training loss, validation loss, and most notably, the anomaly threshold, are increased when increasing the predictive window.

Lastly, observing Table 5.1, it is evident that the validation loss is lower than the training loss for all of the models. Note that this could be misleading since the training loss is calculated as the rolling average of the loss for the past hundred sequences while the validation loss is calculated only at the end of each epoch. Following the training loss, per batch, presented in Figure 5.1 we see that the training and validation loss is essentially the same. Additionally, we can see a quick convergence of the training and validation loss, which indicates that the amount of epochs is sufficient to reach a minimum.

Forecasting result DeepAnT					
p_w	h_w	n_w	MAE		τ
			Train	Validation	
Production part 1					
1	60	9	0.0275	0.0241	0.2635
1	30	9	0.0275	0.0275	0.2635
1	60	100	0.0327	0.0290	0.2793
1	30	100	0.0293	0.0259	0.2732
1	60	200	0.0350	0.0315	0.2923
1	30	200	0.0310	0.0280	0.2749
10	60	9	0.0357	0.0316	0.7522
10	30	9	0.0376	0.0335	0.7670
10	60	100	0.0371	0.0328	0.7753
10	30	100	0.0393	0.0350	0.7891
10	60	200	0.0384	0.0351	0.8023
10	30	200	0.0411	0.0383	0.8370
Production part 2					
1	60	9	0.0302	0.0265	0.2663
1	30	9	0.0264	0.0224	0.2540
1	60	100	0.0302	0.0264	0.2622
1	30	100	0.0264	0.0222	0.2497
1	60	200	0.0308	0.0271	0.2611
1	30	200	0.0310	0.0280	0.2749
10	60	9	0.0340	0.0297	0.7121
10	30	9	0.0345	0.0304	0.7213
10	60	100	0.0342	0.0308	0.7266
10	30	100	0.0354	0.0314	0.7281
10	60	200	0.0365	0.0324	0.7447
10	30	200	0.0377	0.0336	0.7572
Production part 3					
1	60	9	0.0270	0.0235	0.2372
1	30	9	0.0240	0.0198	0.2263
1	60	100	0.0280	0.0249	0.2439
1	30	100	0.0252	0.0210	0.2310
1	60	200	0.0284	0.0248	0.2420
1	30	200	0.0259	0.0215	0.2353
10	60	9	0.0311	0.0272	0.6511
10	30	9	0.0321	0.0285	0.6583
10	60	100	0.0327	0.0287	0.6711
10	30	100	0.0341	0.0298	0.6848
10	60	200	0.0335	0.0293	0.6705
10	30	200	0.0341	0.0299	0.6899

Table 5.1: MAE loss and anomaly threshold (τ) from each of the trial setups. Note that **production part n** refers to the results on the specific part n produced by the manufacturing machine, p_w is the predictive window, h_w is the history window, and n_w is the window of cycles before failures excluded from training.

5.1.2 Anomaly detection

Table 5.2 demonstrates the results of the anomaly detection task. We can see that certain results hold independent of the production part and tool failure investigated. To begin, we can see that the recall for N is 0.95 in every case. Considering that the threshold of anomalies was set to the 95-quantile of the anomaly scores in the training set, it is evident that the amount of anomalies is very similar in the training and testing set of N. Furthermore, the precision for N is decreased while the precision for F is increased following a larger value of n_w. This is trivial since a larger n_w decreases the amount of data belonging to N, causing a smaller amount of false positives, or anomalies in the normal working condition. Thereby, the most reliable parameter setup will mainly be based on the recall. Lastly, since the result varies depending on the production part investigated for F, the discussion will be done separately for each. Before this, the models that are the best in separating N and F in terms of anomalies will be further investigated using confusion matrix and histograms of anomaly scores. This is to further investigate research question 2, where we want to answer if there indeed is a noticeable difference between the anomaly distribution of N and F.

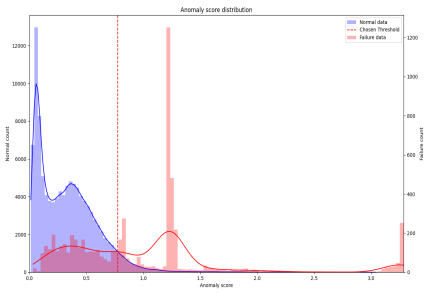
5.1.2.1 Production part 1

In Table 5.2, we see the classification metrics for N and F for tool failures T505 and T151 belonging to production part 1. To begin, the anomaly distribution is affected by the parameter values of p_w, h_w, and n_w. For some of the model setups the eight production cycles leading up to failures reach a recall score above 0.5 for T505 and 0.2 for T151. This means that over half of the predictions are classified as anomalies for F belonging to T505 and over a fourth of the predictions for F belonging to T151 is classified as anomalies. In general, for the multi-step prediction setup, meaning when p_w has the value of 10, and for a larger history window, meaning when h_w has the value of 60, we reach a larger recall for F. With the combination of a smaller history window and single-step predictions we can see that the recall score is below 0.1 for both tool failures, meaning that less than 10 % of the time steps investigated are labeled as anomalies. Lastly, we can see that a larger n_w leads to more anomalies being uncovered before failures for both tools are investigated. T505 reached the highest recall for F with n_w being 100 and T151 reached the highest recall for F with n_w being 200. Thereby demonstrating a performance increase following a increased amount of production cycles pre-failure excluded from training.

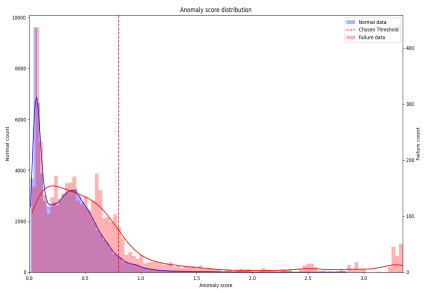
DeepAnT anomaly detection												
(p_w, h_w, n_w)	Precision				Recall				F1-score			
	N	F_{tool1}	N	F_{tool2}	N	F_{tool1}	N	F_{tool2}	N	F_{tool1}	N	F_{tool2}
Production part 1, tool1: T505, tool2: T151												
(1,60,9)	0.98	0.24	0.98	0.07	0.95	0.44	0.95	0.15	0.96	0.31	0.96	0.09
(1,30,9)	0.96	0.01	0.98	0.03	0.95	0.02	0.95	0.06	0.96	0.01	0.96	0.04
(1,60,100)	0.97	0.33	0.97	0.1	0.95	0.45	0.95	0.14	0.96	0.38	0.96	0.11
(1,30,100)	0.95	0.08	0.97	0.08	0.95	0.09	0.95	0.11	0.95	0.08	0.96	0.09
(1,60,200)	0.96	0.40	0.95	0.13	0.95	0.45	0.95	0.14	0.95	0.42	0.95	0.13
(1,30,200)	0.94	0.11	0.95	0.10	0.95	0.09	0.95	0.12	0.94	0.10	0.95	0.11
(10,60,9)	0.98	0.28	0.98	0.08	0.95	0.56	0.95	0.19	0.96	0.37	0.96	0.11
(10,30,9)	0.98	0.25	0.98	0.08	0.95	0.50	0.95	0.19	0.96	0.33	0.96	0.12
(10,60,100)	**0.98**	**0.37**	0.98	0.08	**0.95**	**0.58**	0.95	0.19	**0.96**	**0.46**	0.96	0.11
(10,30,100)	0.97	0.35	0.97	0.13	0.95	0.53	0.95	0.20	0.96	0.42	0.96	0.15
(10,60,200)	0.97	0.44	**0.97**	**0.15**	0.95	0.55	**0.95**	**0.25**	0.96	0.49	**0.96**	**0.19**
(10,30,200)	0.96	0.40	0.96	0.16	0.95	0.49	0.95	0.20	0.96	0.44	0.95	0.18
Production part 2, tool1: T606, tool2: T1228												
(1,60,9)	0.94	0.18	0.99	0.04	0.95	0.15	0.95	0.13	0.94	0.16	0.97	0.06
(1,30,9)	0.93	0.14	0.99	0.03	0.95	0.11	0.95	0.09	0.94	0.12	0.97	0.04
(1,60,100)	0.88	0.30	0.97	0.08	0.95	0.14	0.95	0.13	0.91	0.20	0.96	0.10
(1,30,100)	0.87	0.15	0.97	0.05	0.95	0.06	0.95	0.08	0.91	0.09	0.96	0.06
(1,60,200)	0.77	0.47	0.94	0.15	0.95	0.14	0.95	0.13	0.85	0.21	0.95	0.14
(1,30,200)	0.76	0.28	0.94	0.10	0.95	0.06	0.95	0.08	0.84	0.10	0.94	0.09
(10,60,9)	0.94	0.24	0.99	0.06	0.95	0.21	0.95	0.23	0.94	0.21	0.97	0.10
(10,30,9)	0.95	0.30	0.99	0.05	0.95	0.29	0.95	0.19	0.95	0.27	0.97	0.08
(10,60,100)	0.90	0.34	**0.97**	**0.11**	0.95	0.18	**0.95**	**0.23**	0.91	0.24	**0.96**	**0.15**
(10,30,100)	**0.92**	**0.43**	0.97	0.10	**0.95**	**0.29**	0.95	0.19	**0.92**	**0.33**	0.96	0.13
(10,60,200)	0.77	0.52	0.95	0.20	0.95	0.17	0.95	0.20	0.85	0.26	0.95	0.20
(10,30,200)	0.79	0.6	0.94	0.17	0.95	0.24	0.95	0.16	0.86	0.34	0.95	0.17
Production part 3, tool1: T404, tool2: T199												
(1,60,9)	0.98	0.03	0.98	0.05	0.95	0.08	0.95	0.12	0.97	0.04	0.96	0.07
(1,30,9)	0.98	0.02	0.98	0.03	0.95	0.05	0.95	0.07	0.97	0.03	0.96	0.04
(1,60,100)	0.98	0.03	0.97	0.07	0.95	0.08	0.95	0.12	0.96	0.05	0.96	0.09
(1,30,100)	0.98	0.03	0.97	0.05	0.95	0.07	0.95	0.09	0.96	0.04	0.96	0.09
(1,60,200)	0.97	0.04	0.96	0.09	0.95	0.07	0.95	0.12	0.96	0.05	0.96	0.10
(1,30,200)	0.97	0.04	0.96	0.06	0.95	0.06	0.95	0.08	0.96	0.05	0.95	0.07
(10,60,9)	0.98	0.06	0.98	0.09	0.95	0.19	0.95	0.22	0.97	0.09	0.96	0.13
(10,30,9)	0.98	0.05	0.98	0.09	0.95	0.16	0.95	0.20	0.97	0.08	0.96	0.12
(10,60,100)	**0.98**	**0.08**	0.97	0.12	**0.95**	**0.19**	0.95	0.22	**0.96**	**0.11**	0.96	0.16
(10,30,100)	0.98	0.08	**0.98**	**0.14**	0.95	0.17	**0.95**	**0.26**	0.96	0.11	**0.96**	**0.18**
(10,60,200)	0.97	0.08	0.96	0.15	0.95	0.14	0.95	0.21	0.96	0.11	0.96	0.18
(10,30,200)	0.97	0.09	0.97	0.15	0.95	0.15	0.95	0.20	0.96	0.11	0.96	0.17

Table 5.2: Testing set precision, recall, and F1-score for the DeepAnT model for N and F. The highlighted models are the ones with highest recall for F.

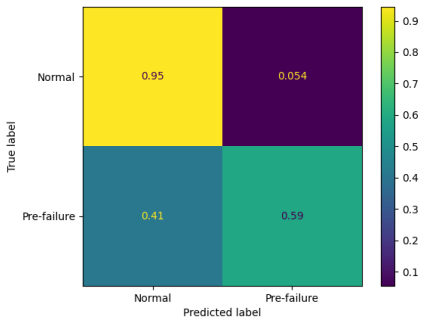
The anomaly distribution of the highlighted models is visualized in Figure 5.2. As to be expected following the resulting metrics, F for T505 shows a clear separation in anomaly score compared to N for T505. The separation for T151 is less evident, however, we still see an increase in higher anomaly scores beyond the threshold value for F as compared to N. To visualize the



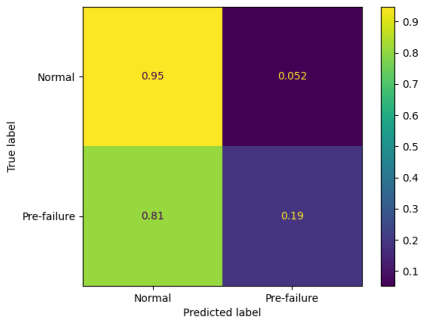
(a) Anomaly score distribution for T505



(b) Anomaly score distribution for T151



(c) Confusion matrix of anomaly predictions in N and F belonging to T505



(d) Confusion matrix of anomaly predictions in N and F belonging to T151

Figure 5.2: Production part 1 anomaly distribution and confusion metric for chosen tool errors.

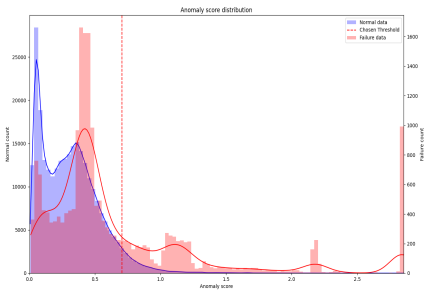
anomaly progression over the eight production cycles preceding and including the failure, Figure 5.5a shows the embedded space from a t-SNE trained on the prediction errors leading up to the failure for T505, and Figure 5.5b shows it for T151. Firstly, for T505, it's easy to conclude that the prediction errors of F show big dissimilarities in prediction errors as compared to the predictions in N. Indeed, almost all production cycles contains time-windows with prediction errors outside the normal working condition cluster.

For T151, the majority of the predictions close to failure show similar characteristics as the predictions for N. The ones that deviate mostly show similar characteristics as anomalies belonging to N. However, we also see prediction errors of F not seen in N in the five and six production cycles before failures.

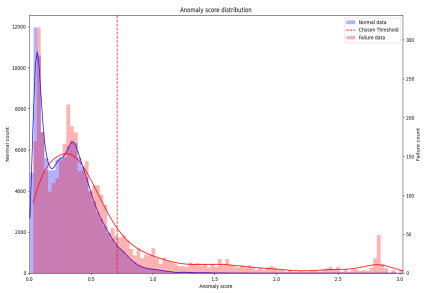
5.1.2.2 Production part 2

In Table 5.2, we see the classification metrics for N and F for tool failures T606 and T1228 belonging to production part 2. In the highlighted rows we see that the recall score of F belonging to T606 received a value of 0.29 and F belonging to T1228 received a value of 0.23. Meaning that 29% of the time steps belonging to the eight production cycles preceding failure for T606 are labeled as anomalies and 22 % of the time steps belonging to the eight production cycles leading up to failure for T1228 were labeled anomalous. A larger predictive window appears to lead to more anomalies being uncovered for both the investigated tool failures. When comparing the results to the smaller predictive window of one, the recall varies between 0.06-0.14, meaning that 6 to 14 % of the time steps preceding failures are labeled as anomalous. Furthermore, there seems to be a discrepancy concerning the optimal history window since more anomalies are uncovered for T606 with h_w being sixty while T1228 uncovers more anomalies with h_w being thirty. Both tool failures investigated do however reach the highest recall for F with n_w being 100.

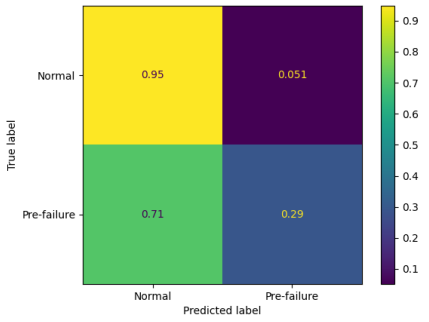
The anomaly distribution of the highlighted models with confusion matrices is presented in Figure 5.3. While the majority of the predicted anomaly scores of F are below the anomaly threshold, we still see spikes in high anomaly score values above the anomaly threshold for both T606 and T1228. To visualize the anomaly progression over the eight production cycles preceding failure, Figure 5.5c and Figure 5.5d shows the t-SNE embedded space of the prediction errors for tool T606 and T1228 respectively. For T616, we can see that several prediction errors of F have a clear separation from N in the same production cycle as the failure occur. In addition, there exist several smaller clusters outside the range of the normal working condition, for example, looking at coordinate $(25, -125)$ of the embedded space. Here, we see a cluster with prediction errors from up to 4 production cycles before failures. We also see a cluster at coordinate $(-125, -55)$, however, occurrences of anomalies from N are seen in this section too.



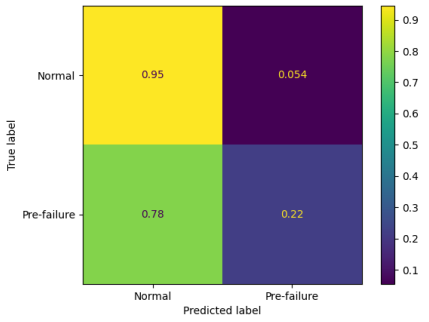
(a) Anomaly score distribution for T606



(b) Anomaly score distribution for T1228



(c) Confusion matrix of anomaly predictions in N and F belonging to T606



(d) Confusion matrix of anomaly predictions in N and F belonging to T1228

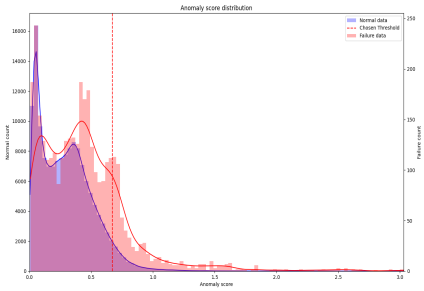
Figure 5.3: Production part 2 anomaly distribution and confusion metric for chosen tool errors.

For T1228, the majority of the prediction errors in F with different characteristics to N are seen in the production cycle containing the failure and one production cycle before failure.

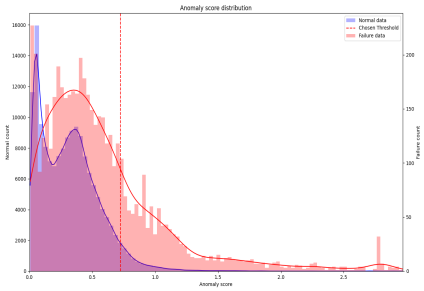
5.1.2.3 Production part 3

In Table 5.2, we see the classification metrics for N and F for tool failures T404 and T199 belonging to production part 3. Similar to the previous tool failures p_w has a significant impact on the anomaly distribution of F. Neither of the model with single steps predictions has a recall for F above 0.08 for T404 and 0.12 for T199, meaning 8 % and 12 % labeled anomalies previous to failure. However, T404 and T199 reaches a maximum recall for F of 0.19 and 0.26 respectively with p_w being ten. Lastly, most anomalies are uncovered with n_w being set to 100.

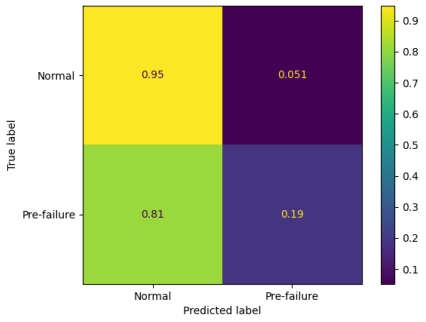
The anomaly distribution of the highlighted models with confusion matrices is visualized in Figure 5.4. For T404, while we still see a larger presence of high anomaly scores compared to the distribution of N, unlike the previous production parts, the anomaly distributions of F seem more similar to the normal data N. This is further demonstrated through the t-SNE visualization in Figure 5.5e. The predictions belonging to F that are separate from N do, in almost all cases, have normal anomalies nearby in the embedded space. Otherwise, there do not seem to be any other deviating clusters formed by F. T199 shows a slightly more dissimilar distribution of anomalies comparing N and F in Figure 5.4. In the t-SNE visualization presented in Figure 5.5f we also see that below four production cycles until failures, several prediction errors in F show distinct prediction errors compared to N.



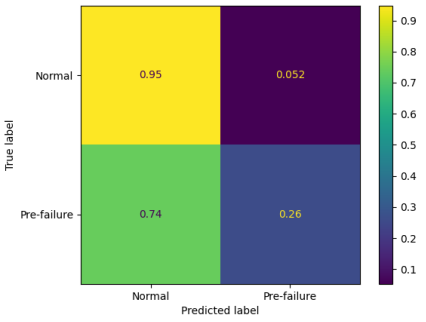
(a) Anomaly score distribution for T404



(b) Anomaly score distribution for T199

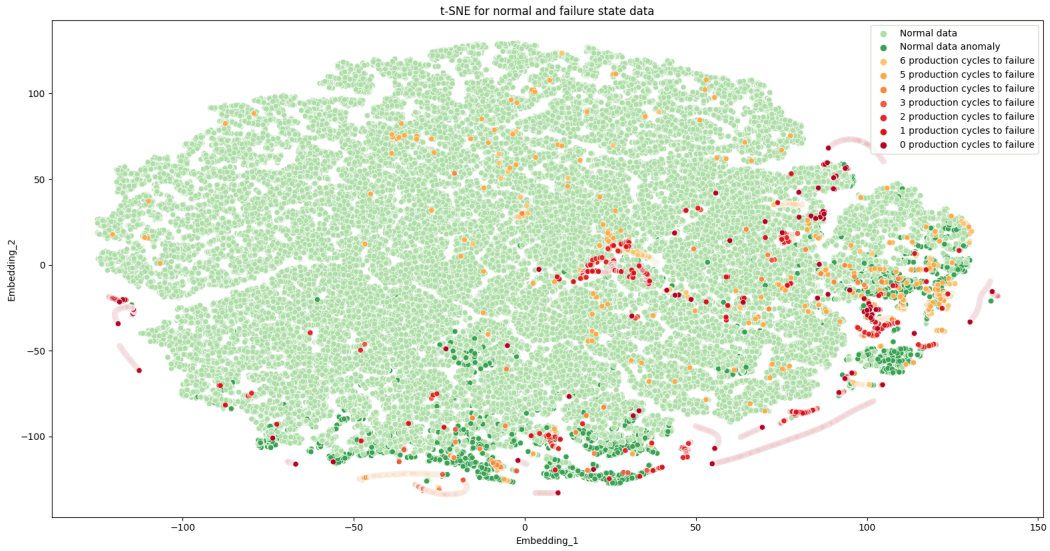


(c) Confusion matrix of anomaly predictions in N and F belonging to T404

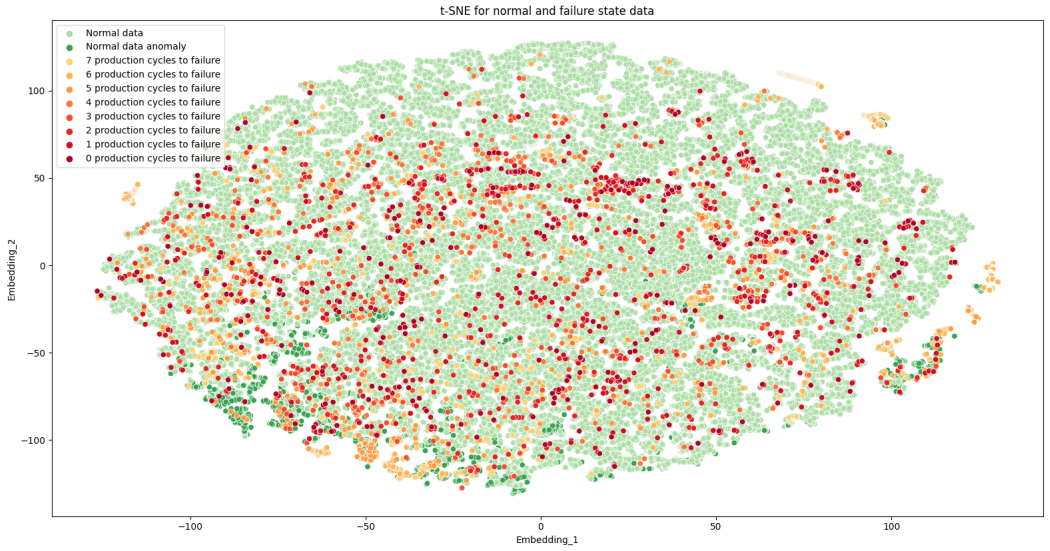


(d) Confusion matrix of anomaly predictions in N and F belonging to T199

Figure 5.4: Production part 3 anomaly distribution and confusion metric for chosen tool errors.

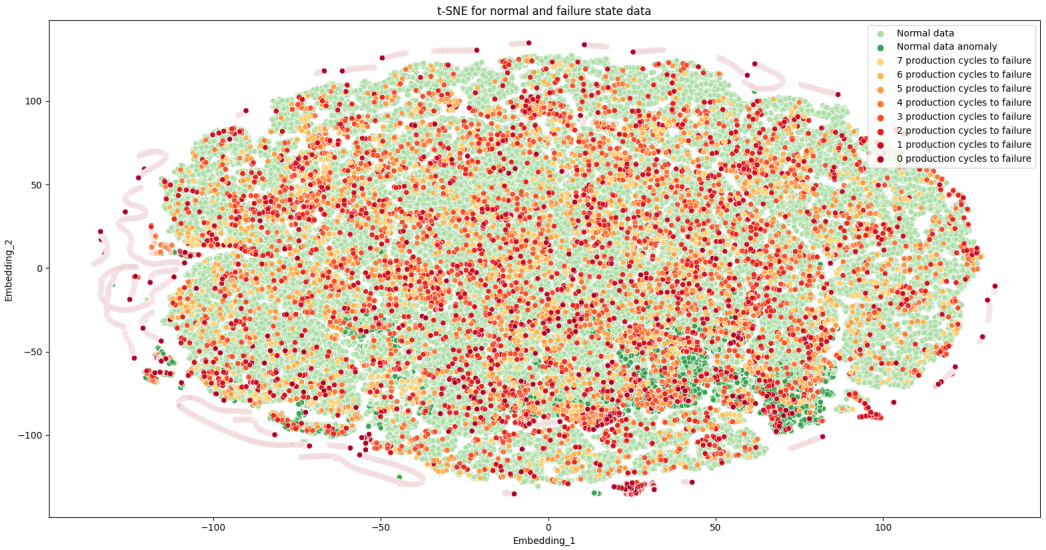


(a) Production part 1, T505

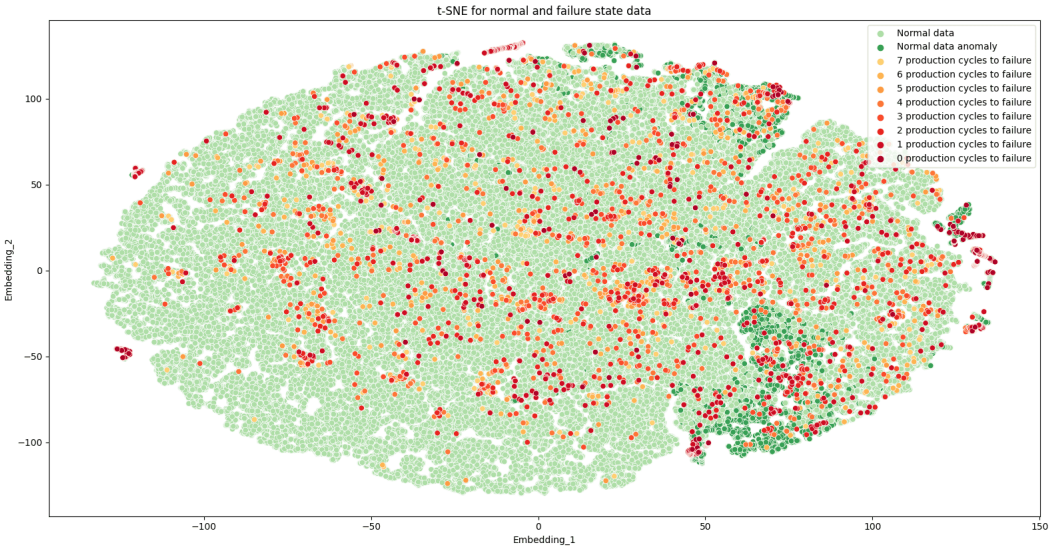


(b) Production part 1, T151

Figure 5.5: T-SNE visualization of the DeepAnt forecasting error for production part 1. The perplexity of the t-SNE was set to 80.

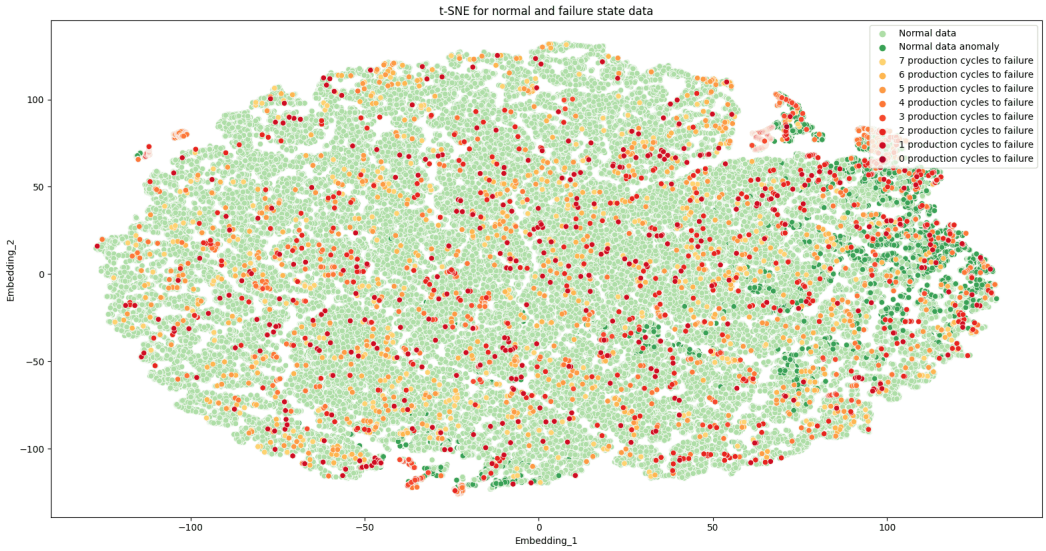


(c) Production part 2, T606

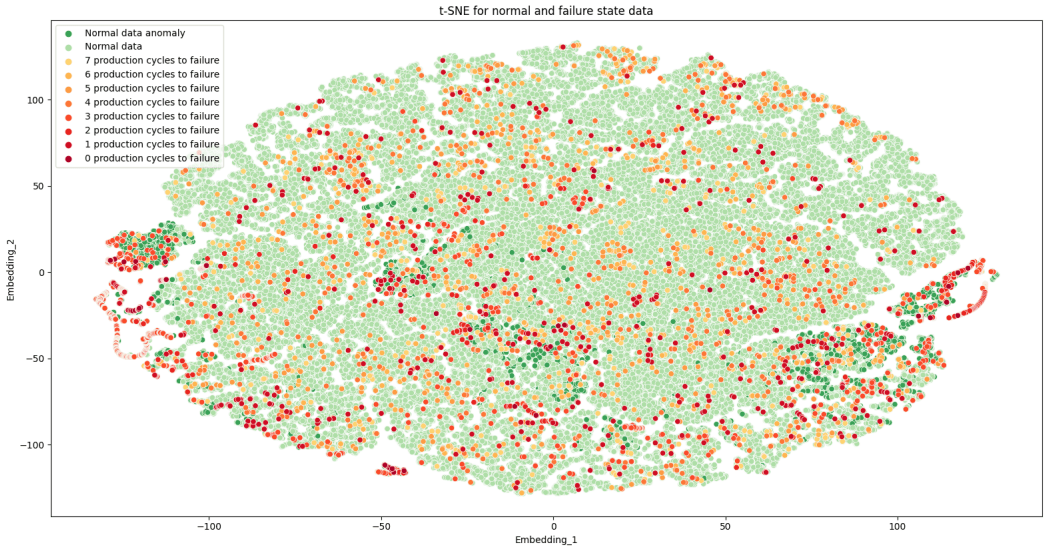


(d) Production part 2, T1228

Figure 5.5: t-SNE visualization of the DeepAnt forecasting error for production part 2. The perplexity of the t-SNE was set to 80.



(e) Production part 3, T404



(f) Production part 3, T199

Figure 5.5: T-SNE visualization of the DeepAnt forecasting error for production part 3. The perplexity of the t-SNE was set to 80.

5.2 ROCKET OCSVM

In this section, we will begin with investigating the cross-validation result for the model selection phase. We will then continue to evaluate anomaly distribution and characteristics based on the chosen models predictions of the testing set for N and F.

5.2.1 Cross validation

To begin, the hyper-parameter tuning with the 5-fold cross-validation scheme is presented in Table 5.3 and Table 5.4. These results will be used as an indication for the parameter selection before the final training and evaluation. The first conclusion that can be made that holds for each production part and tool failure is concerning the recall of N. We see that the performance on labeling N as non-anomalous generalizes well for the validation data. The mean recall for N is never below 0.984 and reaches the limit of 0.01 % errors set for the training set through the parameter τ in many instances. It is worth stating that the sigmoidal and polynomial kernels are superior for reaching a higher recall score for N. This is expected following the reasoning in Section 4.2.2. Indeed, since we are training the OCSVM in a semi-supervised setup with the assumption that all the data belongs to one class a higher recall might be a result of a less restrictive hyperplane separation calculated from the kernel function. When observing the mean recall score of F, we can also see that RBF is generally better for separating the normal and failure data.
To continue with the discussion, since the result for F varies between the different production parts and tool failures we will present the results separately for each production parts.

OCSVM cross validation				
Kernel	l	n_w	$Recall_\sigma$	
			N	F
Production part 1: tool T505				
Sig	232	9	0.990	0.667
Sig	232	100	0.990	0.667
Sig	232	200	0.986	0.667
Rbf	232	9	0.990	0.667
Rbf	232	100	0.984	0.667
Rbf	232	200	0.985	0.667
Poly	232	9	0.990	0.667
Poly	232	100	0.990	0.667
Poly	232	200	0.985	0.667
Production part 2: tool T606				
Sig	96	9	0.990	0.364
Sig	96	100	0.989	0.382
Sig	96	200	0.987	0.364
Rbf	96	9	0.986	0.364
Rbf	96	100	0.988	0.368
Rbf	96	200	0.987	0.364
Poly	96	9	0.990	0.364
Poly	96	100	0.989	0.382
Poly	96	200	0.990	0.364
Production part 3: tool T404				
Sig	90	9	0.990	0.111
Sig	90	100	0.989	0.222
Sig	90	200	0.990	0.222
Rbf	90	9	0.988	0.155
Rbf	**90**	**100**	**0.989**	**0.4**
Rbf	90	200	0.986	0.378
Poly	90	9	0.990	0.011
Poly	90	100	0.989	0.133
Poly	90	200	0.990	0.133

(a) l set to a half of minimum failure production cycle length

OCSVM cross validation				
Kernel	l	n_w	$Recall_\sigma$	
			N	F
Production part 1: tool T505				
Sig	116	9	0.990	0.454
Sig	116	100	0.989	0.455
Sig	116	200	0.990	0.454
Rbf	116	9	0.988	0.636
Rbf	**116**	**100**	**0.988**	**0.727**
Rbf	116	200	0.987	0.655
Poly	116	9	0.990	0.455
Poly	116	100	0.989	0.455
Poly	232	200	0.990	0.455
Production part 2: tool T606				
Sig	48	9	0.989	0.529
Sig	48	100	0.989	0.529
Sig	48	200	0.989	0.517
Rbf	48	9	0.989	0.517
Rbf	**48**	**100**	0.989	**0.529**
Rbf	48	200	0.987	0.529
Poly	48	9	0.990	0.529
Poly	48	100	0.989	0.517
Poly	48	200	0.989	0.517
Production part 3: tool T404				
Sig	45	9	0.990	0.111
Sig	45	100	0.989	0.111
Sig	45	200	0.989	0.111
Rbf	45	9	0.988	0.232
Rbf	45	100	0.988	0.222
Rbf	45	200	0.987	0.222
Poly	45	9	0.990	0.11
Poly	45	100	0.990	0.111
Poly	45	200	0.989	0.111

(b) l set to a third of minimum failure production cycle length

Table 5.3: The resulting mean recall ($Recall_\sigma$) for pre-failure production cycles and normal production cycles over each of the 5 cross validation splits. Kernel is the kernel-function used during OCSVM training, τ, meaning the upper bound set for the percentage of false positives, is set to 0.01. n_w is the amount of cycles before failures excluded from N. The highlighted rows indicate the hyper-parameter setups chosen for evaluation.

OCSVM cross validation				
Kernel	l	n_w	$Recall_\sigma$	
			N	F
Production part 1: tool T151				
Sig	243	9	0.989	0.0
Sig	243	100	0.990	0.0
Sig	243	200	0.989	0.0
Rbf	243	9	0.987	0.0
Rbf	243	100	0.987	0.125
Rbf	243	200	0.986	0.225
Poly	243	9	0.989	0.0
Poly	243	100	0.990	0.0
Poly	243	200	0.989	0.0
Production part 2: tool T1228				
Sig	220	9	0.990	0.0
Sig	220	100	0.989	0.0
Sig	220	200	0.984	0.0
Rbf	**220**	**9**	**0.985**	**0.3**
Rbf	220	100	0.978	0.25
Rbf	220	200	0.978	0.25
Poly	220	9	0.990	0.0
Poly	220	100	0.989	0.0
Poly	96	200	0.989	0.0
Production part 3: T199				
Sig	90	9	0.990	0.182
Sig	90	100	0.989	0.181
Sig	90	200	0.990	0.181
Rbf	90	9	0.989	0.0911
Rbf	**90**	**100**	**0.987**	**0.273**
Rbf	90	200	0.987	0.2
Poly	90	9	0.990	0.182
Poly	90	100	0.989	0.182
Poly	90	200	0.989	0.182

(a) l set to a half of minimum failure production cycle length

OCSVM cross validation				
Kernel	l	n_w	$Recall_\sigma$	
			N	F
Production part 1: T151				
Sig	120	9	0.990	0.5
Sig	120	100	0.990	0.5
Sig	120	200	0.986	0.5
Rbf	120	9	0.987	0.5
Rbf	**120**	**100**	**0.984**	**0.75**
Rbf	120	200	0.983	0.75
Poly	120	9	0.990	0.5
Poly	120	100	0.990	0.5
Poly	120	200	0.986	0.5
Production part 2: T1228				
Sig	110	9	0.990	0.0
Sig	110	100	0.990	0.0
Sig	110	200	0.988	0.0
Rbf	110	9	0.989	0.0
Rbf	110	100	0.987	0.029
Rbf	110	200	0.987	0.03
Poly	110	9	0.989	0.0
Poly	110	100	0.990	0.0
Poly	110	200	0.989	0.0
Production part 3: T199				
Sig	45	9	0.990	0.14
Sig	45	100	0.990	0.1
Sig	45	200	0.989	0.1
Rbf	45	9	0.989	0.045
Rbf	45	100	0.989	0.2
Rbf	45	200	0.989	0.03
Poly	45	9	0.990	0.14
Poly	45	100	0.990	0.136
Poly	45	200	0.989	0.05

(b) l set to a third of minimum failure production cycle length

Table 5.4: The resulting mean recall ($Recall_\sigma$) for pre-failure production cycles and normal production cycles over each of the 5 cross validation splits. Kernel is the kernel-function used during OCSVM training, τ, meaning the upper bound set for the percentage of false positives, is set to 0.01. n_w is the window of cycles before failures excluded from training. The highlighted rows indicate the hyper-parameter setups chosen for evaluation.

5.2.1.1 Production part 1

Observing the results for production part 1 of Table 5.3, it is evident that for T505, with a large time-window, the parameter n_w has no impact on the mean recall for F. Conversely, a large n_w seems to lead to a lower recall score for N without affecting the recall for F. However, when decreasing the window-size, n_w seems to have a more noticeable impact on recall of F, showing the highest number of uncovered anomalies when n_w is set to 100.
Investigating Table 5.4, we can see a noticeable decrease in recall for F belonging to T151 with a larger window-size. The best performing kernel function based on the mean recall of F was RBF, it was also the only kernel function showing a increase in mean recall for F when using a larger n_w.

5.2.1.2 Production part 2

Observing the results for production part 2 of Table 5.3, one can conclude that for T161 the parameter n_w seems to have a noticeable impact on the recall score when using a larger time window l. Independent of the kernel function the highest mean recall for F was always attained with n_w being 100. For the smaller time window, this pattern is not as visible since the only kernel function that benefits from a larger n_w is RBF. We can also see that we had equal performance when using the sigmoidal kernel with n_w set to 9 and for RBF with n_w set to 100. Since RBF outperformed the other kernel functions for production part 1, RBF was prioritized for continued testing.

Observing the results for production part 2 of Table 5.4, for T1228, RBF is the only kernel that uncovers any anomalies in F independent of the time window. One can also conclude that the optimal value for n_w differed depending on the time window l investigated. Nonetheless, the highest recall for F was uncovered with a larger window size and in this instance, n_w being set to 9 gave the highest mean recall for F.

5.2.1.3 Production part 3

Observing the results for production part 3, we begin with investigating the recall for T404 in Table 5.3. One can conclude that n_w seems to have a great impact on the number of anomalies detected in F with the larger time window l. Increasing n_w from 9 to 100 increased the mean recall of F by 30 % for the RBF kernel. A slight increase is also shown for the Sigmoidal and Polynomial kernels. Additionally, in almost every case using a smaller time

window decreases the mean recall of F.
In Table 5.4 we see the result for T199. We can conclude that the sigmoidal and polynomial kernels are negatively affected by an increase in n_w for both time windows l. However, RBF sees a vast increase in recall for F with n_w being set 100 independent of l. It also reaches the highest mean recall for F with a larger time window chosen.

5.2.2 Evaluation of testing set

Table 5.5 shows the evaluation of the testing set with the parameter setups chosen from the 5-fold cross-validation for each tool failure of each production part. To further demonstrate the anomaly distribution and comparing N and F, we can observe the true values over the rows in confusion matrices presented in Figure 5.7. To begin, we can see that N reaches a recall of 99 % for almost all production parts. Knowing that the threshold τ was set to 0.01, we can see that this threshold generalizes to the testing data belonging to N. The only exception is T1228 which received a recall of 0.96. Nonetheless, we can conclude that the number of anomalies uncovered in the normal training and testing sets are almost always equal. Following this decreased number of false positives, we can also see that the precision for F has shown a large increase as compared to the DeepAnt model. This means that, especially considering the imbalance of the data-set, ROCKET OCSVM have a significantly smaller false positive to true positives ratio compared to DeepAnT, ranging from 33 % to 89 %. Indeed, we also see a increase in recall for F compared to DeepAnt. The only exception being T199 which showed a decrease from 26 % to 17 % uncovered anomalies in F with the ROCKET OCSVM setup.
However, while a increase in anomalies is seen previous to all tool failures, the increase differs significantly between the different tools.

OCSVM testing result						
Precision		Recall		f1-score		Production part
N	F	N	F	N	F	
0.98	0.75	0.99	0.68	0.98	0.71	Part 1, T505
0.92	0.89	0.99	0.49	0.95	0.63	Part 2, T161
0.98	0.41	0.99	0.22	0.98	0.29	Part 3, T404
0.96	0.58	0.98	0.35	0.97	0.44	Part 1, T151
0.96	0.33	0.96	0.31	0.96	0.32	Part 2, T1228
0.98	0.37	0.99	0.17	0.98	0.23	Part 2, T199

Table 5.5: Testing set precision, recall, and f1-score for the models selected from the 5-fold-cross-validation.

5.2.3 T-SNE Visualization of anomalies

To get a further understanding of the anomalies, a T-SNE visualization of the ROCKET features used in the anomaly detection task for each tool failure is shown in Figure 5.6. Before presenting the results for each production part, we can see that the time window chosen affects the amounts of clusters belonging to N in the embedded space. Interestingly, the clusters seem to represent the different parts of the production cycle process belonging to the specified time window. Indeed, with a large time window chosen, we get fewer ROCKET features for each production cycle, each with characteristics over a larger time frame, which therefore forms a smaller amount of clusters. A small time window results in a larger amount of ROCKET features per production cycle, capturing characteristics over smaller time frames of the production process and resulting in a larger amount of clusters.

5.2.3.1 Production part 1

For production part 1, observing Figure 5.6a, we can see that for T505 N form 4 distinct clusters. Knowing that the time window was chosen as 116, and the normal production cycle length is around 7 minutes, this further supports the previous reasoning in Section 5.2.3 in regards to the clusters belonging to different parts of the production cycles. Furthermore, even though not every time window belonging to F is classified as anomalies, we see that neither one is centered in any of the clusters formed by N. Interestingly, already at production cycle 7 before failure, we can see deviating time windows that are classified as anomalies by the OCSVM.
Observing Figure 5.6b, for T151, the majority of the predicted anomalous time windows that deviate from the predicted anomalies of N are seen in 5,6, and 7 production cycles before failure. Interestingly, the amount of predicted anomalous time windows closer to the failure decreased, only showing one labeled anomalous time-step in the three production cycles previous to failure.

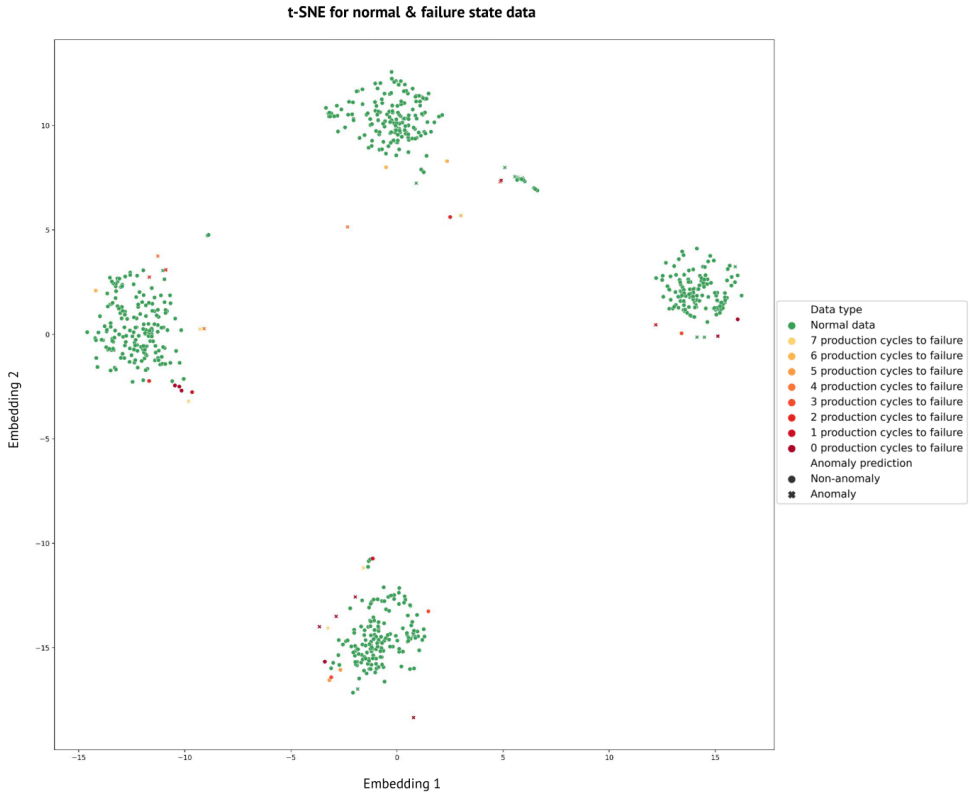
5.2.3.2 Production part 2

For production part 2, in Figure 5.6c, we can see that for T161, there are a larger amount of clusters formed by N, which is likely a result of the smaller time window of 48. Furthermore, in regards to F, there seem to be no labeled anomalous segments 5 production cycles or above the investigated failure. The earliest anomaly is seen 4 production cycles before failure, however, the majority of the predicted anomalies exist in the same production cycle as the

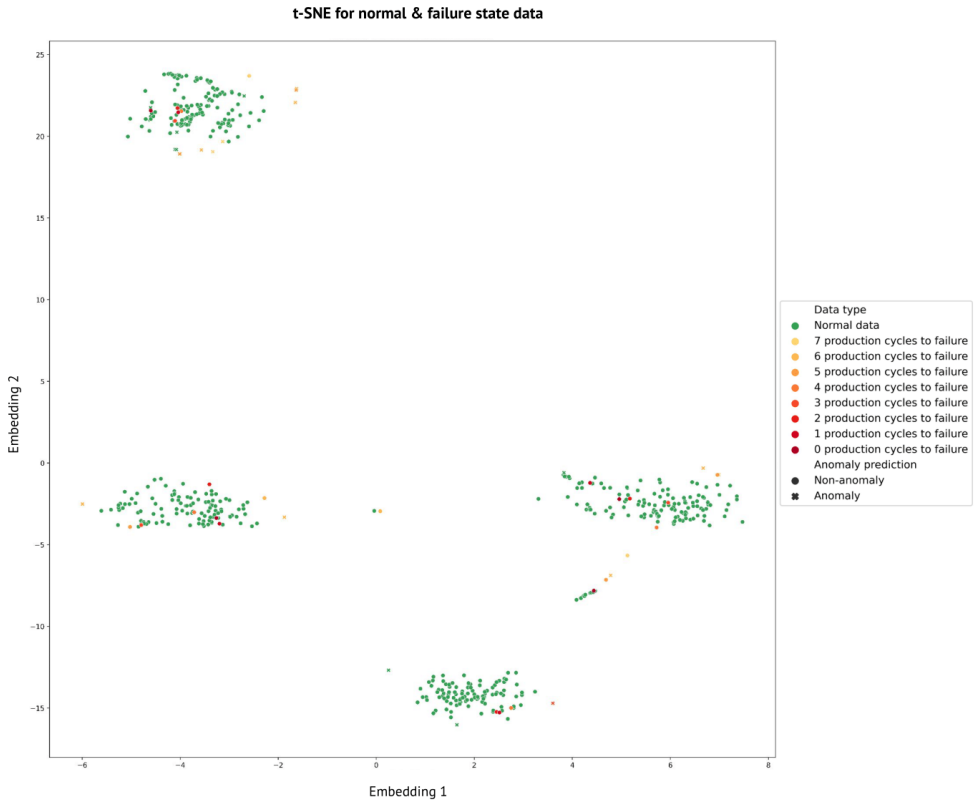
failure, with a large cluster forming outside of the normal working state. Small deviating clusters that include the majority of the production cycles in F are also formed, for example around the area of coordinate $(10, 13)$. However, neither of these is labeled anomalous by the OCSVM.
Observing Figure 5.6d, we can see that for T1228, the larger time window selected from the cross-validation results in two clusters for N. Furthermore, the only production cycle that shows predicted anomalous time windows that are different from the normal working condition is the production cycle previous to failure. However, several non-predicted anomalous time-window in 4, 5, and 6 production cycles before failure can be observed at the edge of the lower cluster formed by N.

5.2.3.3 Production part 3

For production part 3, following the visualization of T404 in Figure 5.6e, we can see that the number of time-windows of F that deviates from the clusters formed by N increase quickly as we get closer to the investigated failure. Indeed, no evident deviations is seen before the five production cycles previous to the failure. However, one can also conclude that several time windows belonging to F that is outside the clusters formed by N in the embedded space are not classified as anomalies by the OCSVM.
Lastly, observing Figure 5.6f, we can conclude that for T199 similarities exist to tool T404. No indication of anomalies is seen above five production cycles before failure, and a sudden increase in deviating time windows is seen in the three production cycles leading up to the failure. Also, as for T404, many of the time windows in F that deviate from the clusters formed by N are not classified as anomalies by the OCSVM.

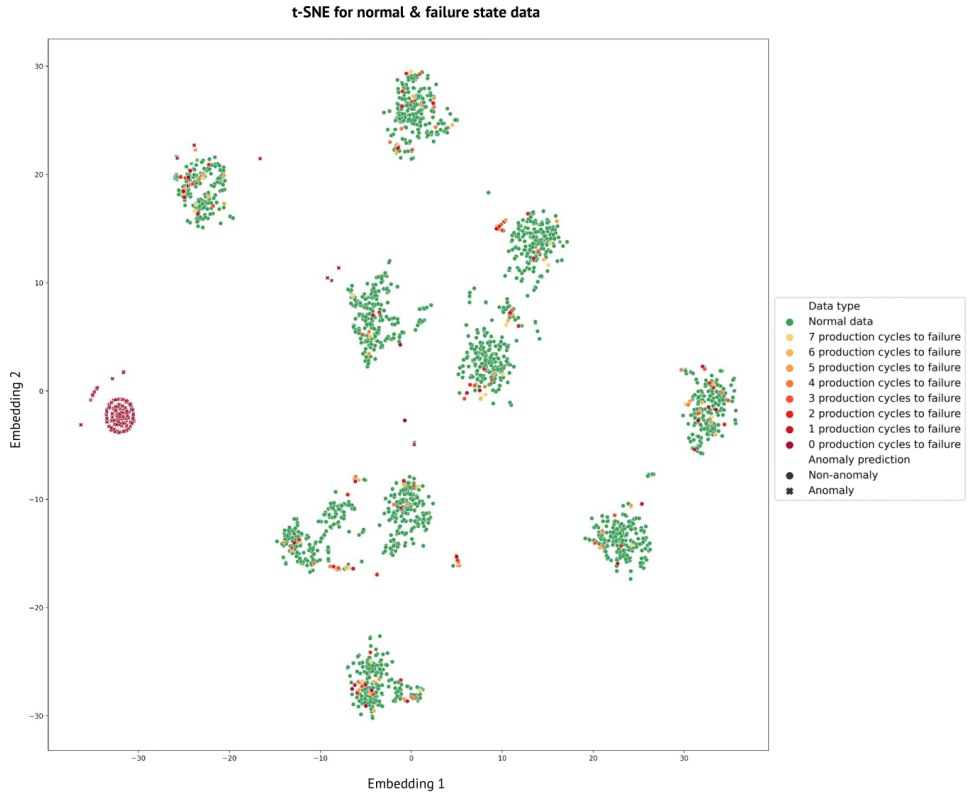


(a) Production part 1, T505

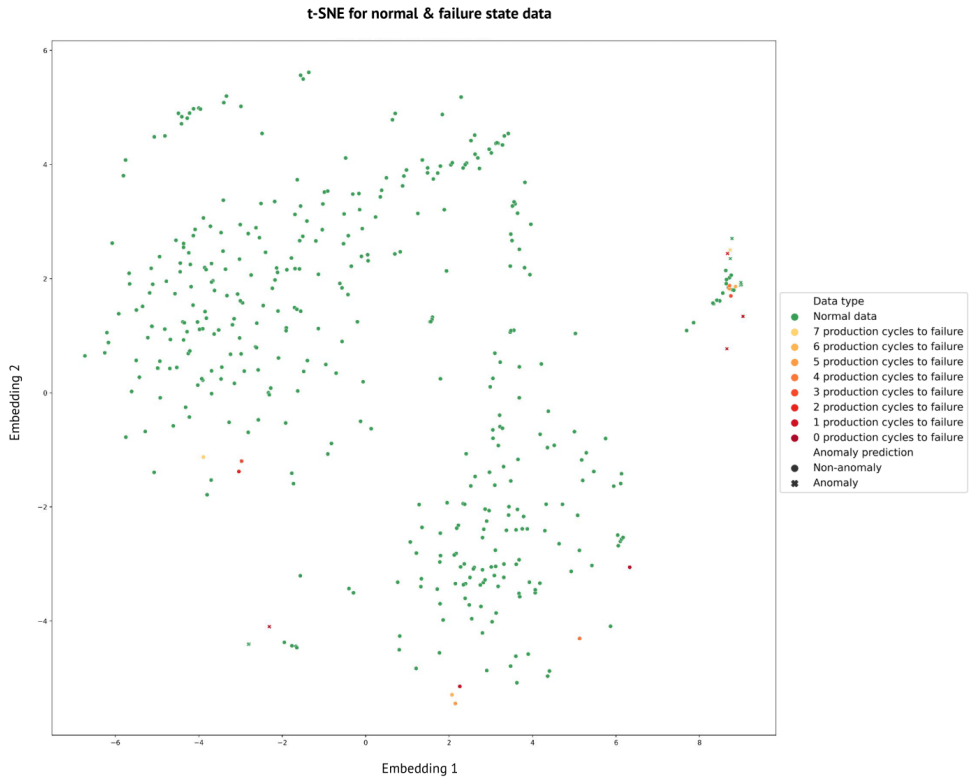


(b) Production part 1, T151

Figure 5.6: T-SNE visualization of the testing data for production part 1. The perplexity of the t-SNE was set to 80.

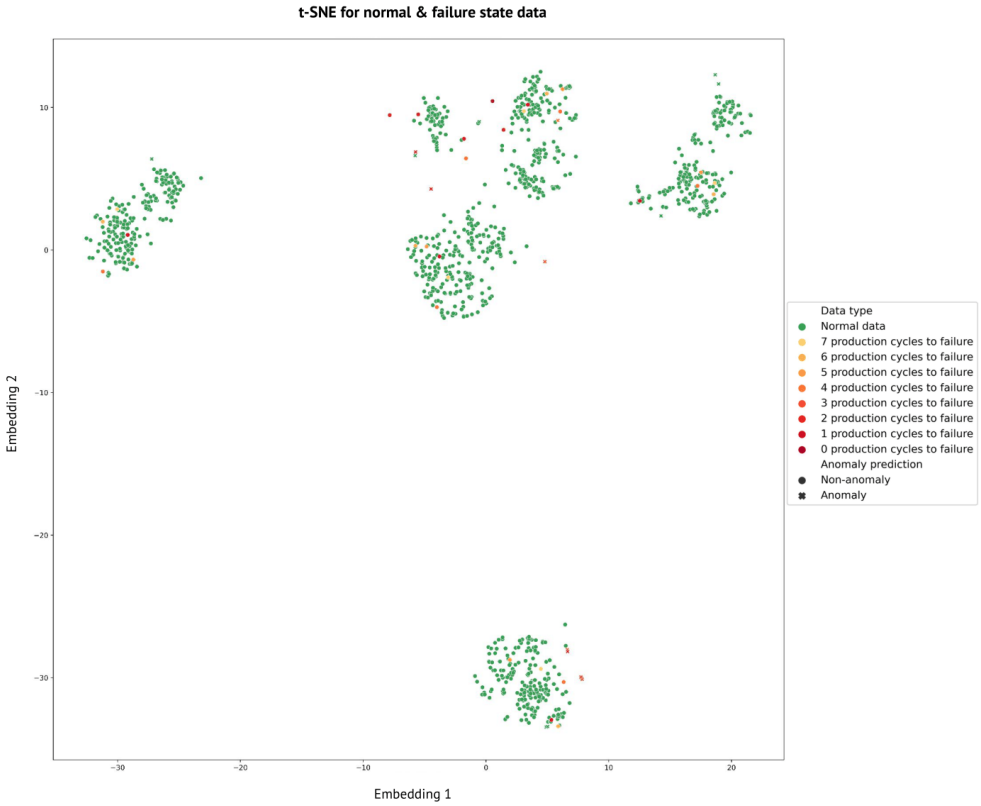


(c) Production part 2, T606

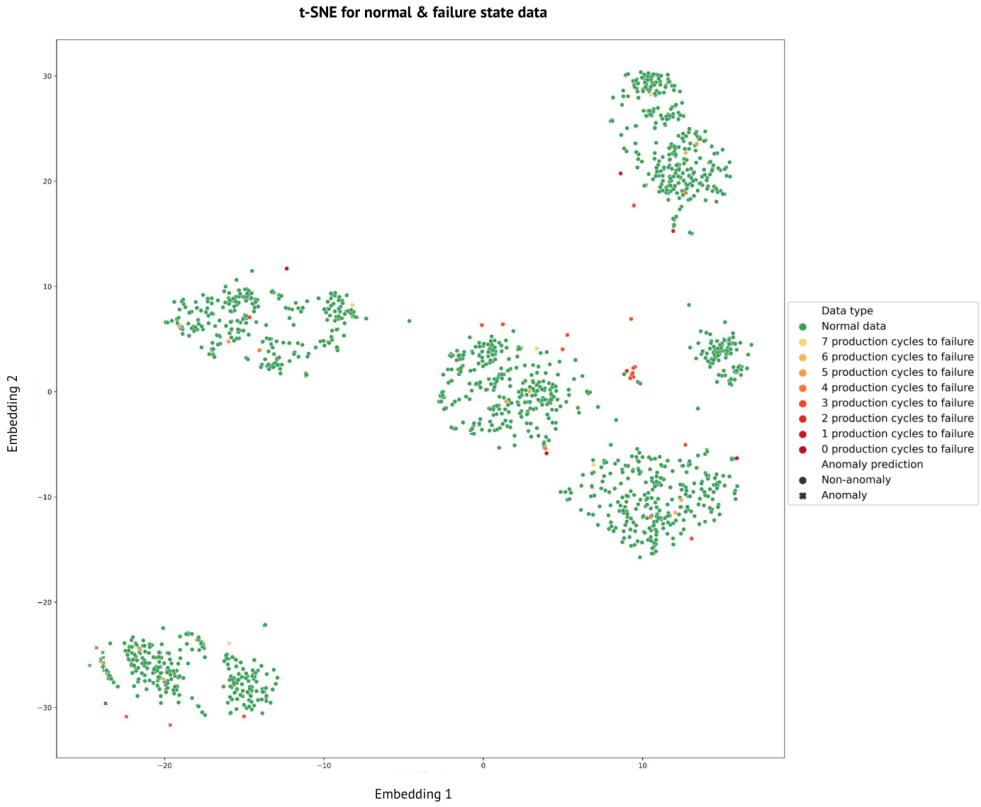


(d) Production part 2, T1228

Figure 5.6: T-SNE visualization of the testing data for production part 2. The perplexity of the t-SNE was set to 80.

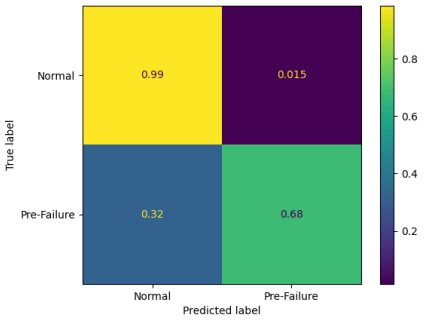


(e) Production part 3, T404

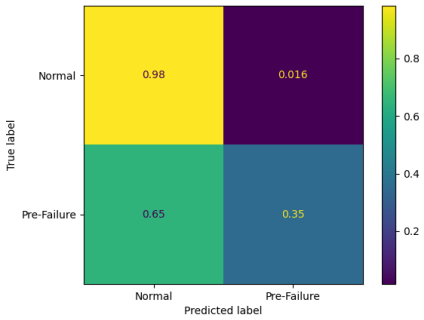


(f) Production part 3, T199

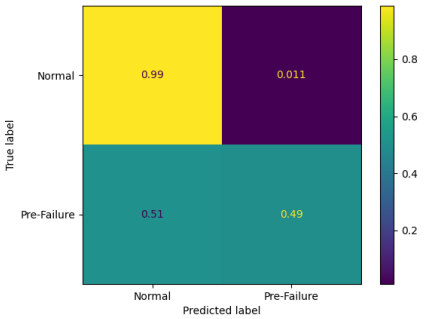
Figure 5.6: T-SNE visualization of the testing data for production part 3. The perplexity of the t-SNE was set to 30.



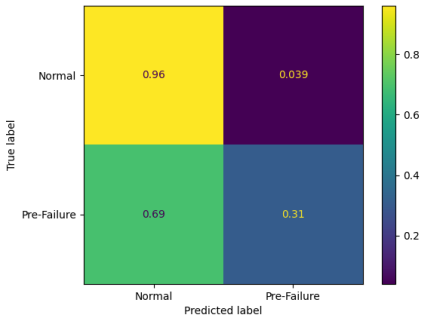
(a) Production part 1, T505



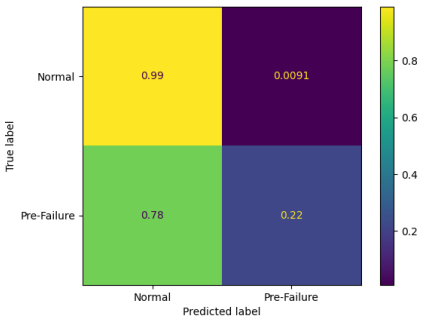
(b) Production part 1, T151



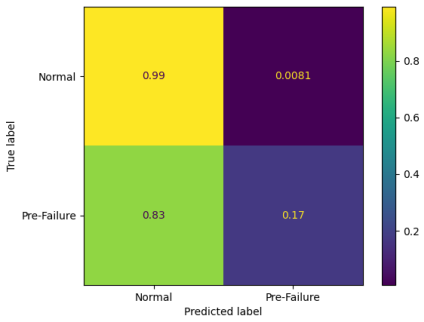
(c) Production part 2, T606



(d) Production part 2, T1228



(e) Production part 3, T404



(f) Production part 3, T199

Figure 5.7: Confusion matrices for each of the chosen models used in the anomaly detection task with ROCKET OCSVM.

Chapter 6

Discussion

Following the results of the experiments, we now want to turn back the focus to research questions and discuss them in the context of the results. We will present a separate discussion for DeepAnT and ROCKET OCSVM then continue with a comparison, discussing their advantages for usage in predictive maintenance. Before this, it is important to note that no results using benchmarks data-sets could be found that use MTConnect data for CNC-machines. Previous research that have investigated CNC-machines have mostly taken use of synthetically injected anomalies constructed through domain knowledge regarding the production process and MTConnect. Also, the data used in this research comes from an currently active CNC-machine that has not been investigated previously. Thereby, no direct comparisons regarding the metrics scores can be done against related works. However, some work have investigated using dimensionality reduction techniques in order to visualize possible anomalies connected to failures using MTConnect data [74], and will serve as a point of comparison in the discussion.

6.1 DeepAnT

We will start with discussing the results of the forecasting ability of the DeepAnT model and continue with the anomaly distribution between normal and pre-failure state data.

6.1.1 Forecasting abilities

Following research question 1, we wanted to investigate the forecasting performance and its dependency on the effect of the parameters n_w, p_w, and

h_w.
Firstly, we can conclude that the forecasting performance was very similar independently of h_w. In some cases, a smaller predictive window reached a lower MAE, for example when comparing rows 3 and 4 of Table 5.1 for production part 1. Possibly, the history window of 30 provides a sufficient amount of information to make the prediction, and the increase to 60 is superfluous. Also, since the production cycles can have stoppages between them and the production part can change over time, we extracted the time windows separately from each production cycle. With the sliding window approach used, predictions can only be made after 60-time steps with h_w being 60. When h_w is 30, predictions can be made after collecting 30-time steps of each production cycle. Therefore, the smaller history window will contain data that the larger predictive window does not have available and, possibly, if this data is more elementary to forecast, this would result in a lower average MAE. In either case, in terms of forecasting performance, 30-time steps seem to suffice.

In regards to the parameter n_w, there was once again little difference between the final MAE. The parameter was added to investigate the prolonged degradation of machine tools. If any long time slight deviations occur in the data before the machine failures, then these would likely affect the forecasting performance. Since the MAE was not affected by the parameter of n_w, this trial seemed to indicate that no long-term deviations were leading up to failures, at least not apparent enough for the CNN to be affected during training and validation. It can be worth noting however that a larger n_w, of course, results in a lower amount of training data, and since we use the same number of epochs for training, the similar performance still indicates that the removed production cycles do not benefit the training. The lesser amount of data is sufficient to converge to a training loss minimum, which could be valuable information since a lower amount of production cycles are necessary to reach satisfying forecasting results.

Lastly, increasing p_w greatly affects the final MAE. Of course, this is to be expected considering that multi-step predictions are generally a harder task than single-step predictions. However, changing p_w can still have a notable effect on the anomaly detection algorithm, which will be discussed in the next section.

6.1.2 Anomaly detection

Following research question 2, we wanted to investigate the distribution of anomalies leading up to failures (F) as compared to the anomaly distribution of the normal working state (N). To begin, we can see that an increase in anomalies is seen in the eight production cycles leading up to failures for all tool failures investigated. The number of anomalies never extended beyond 5 % for N, while for F, the number of anomalies belonging to the eight production cycles before failure mostly varied between 19 and 29 % anomalies. A even larger amount could also be seen for tool failure T151 that reached 58 % anomalies.

Following research question 4, we wanted to discuss how early anomalous segments seem to appear that could be connected to failures. Following the t-SNE visualizations and the result of section 5.1.2, we can once again conclude that the result deviates dependent on the tool failures. For the tool failure of T505 of production part 1, the anomalous segments seem to be present in all eight production cycles leading up to failure, indicating a possible gradual degradation of the machine tools.
For other tool failures, like T606, there seem to be only anomalous segments in the production cycle of the failure, likely indicating that there is a sudden error in the production process. Similarly, the tool failure of T199 also showed several production cycles with normal working characteristics, followed by sudden anomalous segments present in the three production cycles leading up to failure. This, once again, could indicate a sudden error in the production process.
In other cases, like T404 of production part 3, all pre-failure predictions have similar characteristics as the normal production cycle predictions, since they are placed nearby in the embedded space. It is therefore hard to conclude if these anomalies are indications of failures or if they are simply normal working conditions that the forecasting model fails to predict with high accuracy. To conclude, how early anomalies connected to failures are visible fluctuates greatly between different tool failures.

While the forecasting ability was not greatly affected by the investigated parameters, the anomaly detection results differed significantly in terms of recall for F. Especially considering the multi-step and single-step models. As presented previously, multi-step prediction gives more reliable results than single-step predictions in regards to capturing changes in amplitude,

frequency, or other characteristics connected to collective anomalies. Possibly, this indicates that the anomalies that do occur previous to failure are collective anomalies. Furthermore, point anomalies can be greatly affected by a pre-processing scheme like the one presented in this report. The initial sensory channels were too volatile to perform a forecasting approach on and the smoothing of the time series, including exponential moving average and log-transformation, was done to uncover the underlying trend and relevant patterns to make possible anomalies stand out. This can affect possible point anomalies connected to failures.
Furthermore, for all investigated tool failures the number of anomalies previous to failures was increased with a larger value of n_w and finding an optimal value with n_w being set to 100. This contradicts the results from the forecasting MAE. One possible reason for this result could be over-fitting to the smaller amount of production cycles trained on. However, then we would likely see noticeable decrease in training loss and a bigger difference in validation and training loss.

Lastly, to compare to related work, Zhang et al. also showed the possibility of detecting and visualizing anomalous segments of MTConnect data using PCA and a separate feature extraction methodology previous to a tool failure [74]. However, this research only focused on one specific tool failure and only investigated one production cycle before the involved failure. Indeed, following this research, there seems to exist possible anomalies even further back than one production cycle for multiple tools. It is important to note however that the embedded space of the t-SNE used for the DeepAnT prediction errors seems to give less intuitive visualization of anomalies. This could possibly be a result of inefficient parameter selection. While the perplexity was increased to separate the densely-distributed prediction errors better, the value might still be to small to visually separate the anomalies from the non-anomalies in a comprehensive manner.

6.2 ROCKET OCSVM

Following research question 3, we wanted to investigate the distribution of anomalies leading up to failures (F) as compared to the anomaly distribution of the normal working state (N), for the ROCKET OCSVM. Firstly, this anomaly detection technique also found an abundance of anomalies in the eight production cycles leading up to failures compared to the normal working state of the machine. For comparison, observing the final recall, we see that N

contained only 1 % labeled anomalies for almost all investigated production part processes. However, in the time windows of F, the number of anomalies ranged between 17 % and 68 % for the investigated tool failures. Once again, a clear increase in anomalies.

Next, for the time window removal, or the value of n_w, following the initial 5-fold-cross validation the optimal value seems to be 100. For almost all investigated tool failures, the combination of RBF-kernel and n_w being set to 100 gave the highest recall for F, the only exception being T1228 for production part 2 which showed the highest performance with n_w being 9. However, we could also see that T1228 reached the lowest recall for N in the testing phase. One possibility is that for the training, validation, and testing data split, a large amount of the data belonging to the 100 production cycles before failures was included in the testing data. The drop in recall during testing would then indicate that there indeed are anomalous segments in these production cycles. This would also explain why the 5-fold cross-validation was not greatly affected by including these production cycles for T1228, but the remaining tool failures were.

To answer research question 4, meaning how far back anomalies occur that could be an indication of future failure, we turn to the t-SNE visualization of Figure 5.6 and the results presented in Section 5.2.3. Similarly to DeepAnt the characteristics of anomalies vary between tool failures. While certain tool failures like T505 show clear deviations from the normal working condition in all eight production cycles leading up to the failure, other failures like T161 only show an abundance of anomalies in the production cycle containing the failure. Once again, we can conclude that the antecedency of anomalies vary for involved tool failures.

Lastly, to compare the results to Zhang et al. [74]. The t-SNE plot for the ROCKET features seems to not only give similar interoperability, possible following from the less-dense distribution of the ROCKET features, but once again finds time windows further back than one production cycle for several tools that could give indications for future failures. Furthermore, the occurrence of separate clusters for different parts of the production process discussed in the results could enable enterprises to investigate the state of the machine during the different time-windows.

6.3 DeepAnT and ROCKET OCSVM comparison

Regarding research question 5, we wanted to answer how the models compared regarding the anomaly detection task and which one is the most reliable for manufacturing enterprises. Firstly, we can see that both models indicate, somewhat, similar amounts of anomalies pre-failures. For example, both methods agreed on the higher anomaly distribution before the tool failure T505 compared to the other tool failures. They both also agreed on an increased amount of anomalies mainly in the same production cycle as the failure for T161. Interestingly, both models also agreed that for T151, anomalous segments are only seen five production cycles or longer before failure and that the time windows closer to the failure contain fewer anomalies. One possibility is the usage of different tools in different production cycles. The tools used in each production cycle can indeed change and the anomalies that do occur may be only connected to the active regions of the tool that later breaks. Possibly, T151 was not active in the four production cycles up until failure which would explain the results of both models. In either case, the similarity of the results from the different methods for several tool failures gives credibility to the results of the separate models. Both models also separated the normal working condition data and failure data optimally with 100 production cycles excluded before failure for almost all tool failures investigated. This indicates that the removed production cycles do contain some sort of noise or anomalies that affects the anomaly detection task. This also answers research question 6 where we can conclude that the optimal value of excluded pre-failure production cycles is 100.
Furthermore, ROCKET OCSVM shows more success in differentiating between the anomaly distribution of N and F, considering the higher recall for both N, but also F when comparing the results. One possible reason for this could be the nature of the anomalies. While DeepAnT looked at smaller time-frames, predicting either one time step or ten-time steps ahead, OCSVM investigates the presence of anomalies over a larger range of time steps. This is further supported by the results and discussion of DeepAnt in Section 6.1 where the multi-step prediction was superior in uncovering anomalies connected to tool failure, indicating that the anomalies present are collective.

To discuss the reliability of both models in a real manufacturing setup, while

DeepAnT was trained entirely in the LUPE setup, the ROCKET OCSVM did use a small portion of the failure time window for feature selection following the ROCKET feature extraction process. In a real manufacturing setup, enabling the usage of a model through a small amount of failure time steps creates a dependency on available labeled failures. Conversely, the LUPE setup does not need this, and might therefore be more practical. Another argument in support of the DeepAnT model is the forecasting approach of anomaly detection. While ROCKET OCSVM can only make predictions on previously collected observations, DeepAnT can make predictions on forthcoming values and label them in real-time. However, it is worth stating that the normalized cycle-time feature used for DeepAnT would not be available for making real-time predictions since it depends on the full production cycle time. It would therefore be necessary to investigate its performance excluding this feature.
Lastly, as described previously for predictive maintenance one of the most important considerations when enabling the anomaly detection method for usage in real manufacturing setups is the number of false positives generated by the model. Any false alarm connected to a false anomaly prediction might halt the production process and thereby reduce both the efficiency of the production process and the reliability of the model. Considering that ROCKET OCSVM showed a higher recall for N, meaning fewer anomalies uncovered in the production cycles that do not lead to failure, while in many cases finding a larger amount of anomalies pre-failure, ROCKET OCSVM seems to be a more optimal choice in a real manufacturing setup.

6.4 Limitations

There are parts of the methodology that needs to be considered in the results. To begin, while it is necessary to incorporate methods for handling missing time stamps, the impact of linear interpolation is not investigated through the results. The tool failures picked in the evaluation were chosen with this in mind and only failures including low amounts of added time-stamps were considered for the evaluation. However, since no investigation regarding the impact of this pre-processing method was done, the results could be affected without our knowledge.
Secondly, only two tools were investigated for each production part process. Further investigation of the remaining tools could be needed to draw more generalized conclusions regarding the presence of anomalies before failures.

Especially considering the feature selection process of ROCKET features. While ROCKET OCSVM shows promising results for separating the normal working condition and pre-failure production cycles through anomalies, the effect of the feature selection process is not investigated entirely. It could be possible that the feature selection method proposed allows us to fit the anomaly detection model very well to the specified failures while failing to generalize to new tool failures not investigated in this research.

6.5 Sustainability

Following the results of this study, there indeed seem to exist indications for future failure in the machine data based on the anomaly distribution of pre-failure and normal working conditions of the machine. While a full predictive maintenance strategy is not realized, this shows that it is possible, and several of the sustainability impacts listed in Section 1.6, could be achieved following the incorporation of predictive maintenance in the system of the CNC machine. This includes the reduction of research usage, reduced downtime of machines, and increased operator safety. However, to further discuss the topic of sustainability from the results, we can begin with discussing economic consequences. It is important to consider that the work was tailored to the data of one CNC machine. Indeed, to test the generalized performance and avoid possible bias, further research could be needed to ensure that the results are fair. Aimlessly using the same methodology for very different production processes might not yield the same results, and the consequent financial investments needed to realize the system could be lost. Indeed, since anomaly detection can be applied in a wide variety of areas, this problem is not limited to the area of manufacturing. Anomaly detection has seen increased interest in security, medical and financial applications in recent years. While the methodology of this work could be extended to these areas, possibly increasing efficiency and economic gain, it is important to consider the data characteristics available to reach optimal performance and avoid redundant financial investments.

Furthermore, in regards to an ecological perspective, while the training of machine learning models can require computational power and thereby result in large energy consumption, the chosen architectures show good parameter efficiency and low training time compared to other architectures like recurrent neural networks. This results in a lower computational cost and therefore lower energy consumption as a result of model training. It is also worth stating

that identifying pre-failure states could help identifying inefficient tool usage since the methodology allowed us to investigate the state of different active tools and different tool failures. Having efficient tool usage results in more efficient production processes and allow manufacturing enterprises to reduce the activity time of machines, resulting in lower energy consumption or carbon emission of the involved machines.

From the social perspective, Section 1.6 already presented the problem of automation of human working tasks, but further points can be made regarding the effect on workplace personnel. There should exist support for the operators in transitioning to a predictive maintenance strategy, for example using explainability, to ensure that they have sufficient skills in dealing with the transition. Once again, this problem is not limited to the area of manufacturing. Creating a dependency on machine learning solutions in society should be controlled since we want to use human domain knowledge and machine learning systems together. To enable this, we need satisfactory human-computer interaction strategies.

6.6 Conclusions

In this work, we have investigated the potential of using anomaly detection as an indicator for future tool failures in an active CNC machine. We proposed two different semi-supervised anomaly detection models for the task, DeepAnT, and ROCKET OCSVM, and evaluated both models based on different parameter setups. For DeepAnT, we concluded that the forecasting performance was not highly dependable on the parameters n_w and history window but saw a high decrease in performance for a larger predictive window. However, for the anomaly detection task, the result was highly dependable on n_w and the predictive window. Indeed, setting n_w to 100 and a larger predictive window of 10 showed the most reliable results for separating the normal and pre-failure data based on anomalies. This shows that manufacturing enterprises need to differentiate between the anomaly detection model's regression abilities and the anomaly detection score. Indeed, for forecasting anomaly detection models such as DeepAnT, the optimal anomaly detection score does not seem to be reached with a lower regression loss.
For ROCKET OCSVM, using the RBF Kernel with n_w being 100 showed the most promising results for separating normal and pre-failure data. Both models thereby agreed that for the majority of the tool failures, optimal performance is reached with 100 production cycles excluded before failure.

This indicates that possible degradation or further anomalies are present pre-failure and could give reasons to explore degradation monitoring based on MTConnect data further.
Both of the models also predict an abundance of anomalies previous to failures for every tool failure investigated when compared to the normal working condition data of the machine. The models also agreed on the characteristics of the anomalies where certain tool failures showed anomalies as far back as eight production cycles and other tool failures only show anomalies in the production cycle containing the failure. Nonetheless, this shows the possibility of using anomaly detection on MTConnect data as a foundation for predictive maintenance. Alarm rules could be based on anomalies to indicate possible future tool failures. If realized, the automatic detection of future failures could reduce the need for preventive or reactive maintenance, machine downtime, resources, and expenses resulting from machine failure.
Lastly, we observed that ROCKET OCSVM achieves a lower false positive rate, meaning a lower number of anomalies uncovered in production cycles that do not lead to failure, while the majority of the tools find an increased amount of anomalies pre-failure. Following the importance of not delivering false alarms to operators during production processes, ROCKET OCSVM seems to be a more reliable alternative for real manufacturing enterprises.

6.7 Future work

Anomaly detection is a great initial step to realize other predictive maintenance strategies in manufacturing. For example, considering that anomalies do seem to be uncovered before failure, one interesting extension that could be of interest to manufacturing companies is the estimation of the remaining useful time (RUL) of machine tools. Previous research has shown how conventional RUL estimators can achieve significant performance increases when being coupled with anomaly detection methods [111]. Of course, while this research investigates the occurrence of anomalies separately for each part production process, this would require going even further in the data separation process and investigating the status of each tool, for each production part, separately. To enable this, the frequency of the data-gathering process might need to be increased since the time-window of the separate tools being used is very low compared to the production cycle length§.
Lastly, the results show promising results for using ROCKET as a feature extraction method for anomaly detection in time series. However, observing the t-SNE visualization of Figure 5.6, there seems to exist deviating time

windows belonging to pre-failure production cycles that are not classified as anomalies by the OCSVM. This could indicate that the performance of the OCSVM is a bottleneck in the anomaly detection task and future research could investigate the performance of other unsupervised anomaly detection algorithms coupled with ROCKET, including LOF, DBSCAN, or Isolation Forrest.

References

[1] M. Bertolini, D. Mezzogori, M. Neroni, and F. Zammori, "Machine learning for industrial applications: A comprehensive literature review," *Expert Systems with Applications*, vol. 175, p. 114820, 2021. doi: https://doi.org/10.1016/j.eswa.2021.114820. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S095741742100261X [Page 1.]

[2] P. Agarwal, M. Tamer, M. H. Sahraei, and H. Budman, "Deep learning for classification of profit-based operating regions in industrial processes," *Industrial & Engineering Chemistry Research*, vol. 59, no. 6, pp. 2378–2395, 2019. [Page 1.]

[3] J. Yu, "Enhanced stacked denoising autoencoder-based feature learning for recognition of wafer map defects," *IEEE Transactions on Semiconductor Manufacturing*, vol. 32, no. 4, pp. 613–624, 2019. [Page 1.]

[4] D. Kim and S. Kang, "Effect of irrelevant variables on faulty wafer detection in semiconductor manufacturing," *Energies*, vol. 12, no. 13, p. 2530, 2019. [Page 1.]

[5] P. Kamat and R. Sugandhi, "Anomaly detection for predictive maintenance in industry 4.0-a survey," in *E3S web of conferences*, vol. 170. EDP Sciences, 2020, p. 02007. [Pages 1, 2, and 12.]

[6] Y. Ran, X. Zhou, P. Lin, Y. Wen, and R. Deng, "A survey of predictive maintenance: Systems, purposes and approaches," *arXiv preprint arXiv:1912.07383*, 2019. [Page 1.]

[7] "A systematic literature review of machine learning methods applied to predictive maintenance," *Computers Industrial Engineering*, vol. 137, p. 106024, 2019. doi: https://doi.org/10.1016/j.cie.2019.106024.

[Online]. Available: https://www.sciencedirect.com/science/article/pii/S0360835219304838 [Page 1.]

[8] T. Wuest, D. Weimer, C. Irgens, and K.-D. Thoben, "Machine learning in manufacturing: advantages, challenges, and applications," *Production & Manufacturing Research*, vol. 4, no. 1, pp. 23–45, 2016. doi: 10.1080/21693277.2016.1192517 [Page 1.]

[9] Z. Welz, "Integrating disparate nuclear data sources for improved predictive maintenance modeling: Maintenance-based prognostics for long-term equipment operation," 2017. [Page 1.]

[10] O. Serradilla, E. Zugasti, J. Ramirez de Okariz, J. Rodriguez, and U. Zurutuza, "Adaptable and explainable predictive maintenance: Semi-supervised deep learning for anomaly detection and diagnosis in press machine data," *Applied Sciences*, vol. 11, no. 16, 2021. doi: 10.3390/app11167376. [Online]. Available: https://www.mdpi.com/2076-3417/11/16/7376 [Page 2.]

[11] L. Stojanovic, M. Dinic, N. Stojanovic, and A. Stojadinovic, "Big-data-driven anomaly detection in industry (4.0): An approach and a case study," in *2016 IEEE International Conference on Big Data (Big Data)*, 2016. doi: 10.1109/BigData.2016.7840777 pp. 1647–1652. [Pages 2 and 5.]

[12] "Nytt-tech," https://nytt-tech.com/, accessed: 2022-03-08. [Page 2.]

[13] "Mtconnect protocol," https://www.mtconnect.org/, accessed: 2022-03-08. [Pages 2 and 38.]

[14] "precima," https://www.precima.se/sv, accessed: 2022-03-08. [Page 2.]

[15] A. M. Mamadjanov, S. M. Yusupov, and S. Sadirov, "Advantages and the future of cnc machines," *Scientific progress*, vol. 2, no. 1, pp. 1638–1647, 2021. [Page 3.]

[16] G. Martinov, R. Pushkov, and S. Evstafieva, "Collecting diagnostic operational data from cnc machines during operation process," in *IOP Conference Series: Materials Science and Engineering*, vol. 709, no. 3. IOP Publishing, 2020, p. 033051. [Page 3.]

[17] J. Downey, D. O'Sullivan, M. Nejmen, S. Bombinski, P. O'Leary, R. Raghavendra, and K. Jemielniak, "Real time monitoring of the cnc process in a production environment-the data collection & analysis phase," *Procedia Cirp*, vol. 41, pp. 920–926, 2016. [Page 3.]

[18] A. Bousdekis, D. Apostolou, and G. Mentzas, "Predictive maintenance in the 4th industrial revolution: Benefits, business opportunities, and managerial implications," *IEEE Engineering Management Review*, vol. 48, no. 1, pp. 57–62, 2019. [Page 3.]

[19] R.-J. Hsieh, J. Chou, and C.-H. Ho, "Unsupervised online anomaly detection on multivariate sensing time series data for smart manufacturing," in *2019 IEEE 12th Conference on Service-Oriented Computing and Applications (SOCA)*, 2019. doi: 10.1109/SOCA.2019.00021 pp. 90–97. [Pages 4 and 6.]

[20] M. Munir, S. A. Siddiqui, A. Dengel, and S. Ahmed, "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 [Pages xi, 4, 5, 20, 28, 29, and 30.]

[21] M. E. Villa-Pérez, M. A. Alvarez-Carmona, O. Loyola-Gonzalez, M. A. Medina-Pérez, J. C. Velazco-Rossell, and K.-K. R. Choo, "Semi-supervised anomaly detection algorithms: A comparative summary and future research directions," *Knowledge-Based Systems*, vol. 218, p. 106878, 2021. [Page 5.]

[22] L. Ruff, R. A. Vandermeulen, N. Görnitz, A. Binder, E. Müller, K.-R. Müller, and M. Kloft, "Deep semi-supervised anomaly detection," *arXiv preprint arXiv:1906.02694*, 2019. [Pages 5 and 17.]

[23] K. Choi, J. Yi, C. Park, and S. Yoon, "Deep learning for anomaly detection in time-series data: Review, analysis, and guidelines," *IEEE Access*, vol. 9, pp. 120 043–120 065, 2021. doi: 10.1109/ACCESS.2021.3107975 [Page 5.]

[24] M. Munir, M. A. Chattha, A. Dengel, and S. Ahmed, "A comparative analysis of traditional and deep learning-based anomaly detection methods for streaming data," in *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, 2019. doi: 10.1109/ICMLA.2019.00105 pp. 561–566. [Page 5.]

[25] Z. Wu and K. Wu, "Prognostic classification based on random convolutional kernel," *arXiv preprint arXiv:2204.04527*, 2022. [Page 5.]

[26] B. Hrnjica and S. Softic, "Explainable ai in manufacturing: a predictive maintenance case study," in *IFIP International Conference on Advances in Production Management Systems*. Springer, 2020, pp. 66–73. [Page 6.]

[27] J. F. Arinez, Q. Chang, R. X. Gao, C. Xu, and J. Zhang, "Artificial Intelligence in Advanced Manufacturing: Current Status and Future Outlook," *Journal of Manufacturing Science and Engineering*, vol. 142, no. 11, 08 2020. doi: 10.1115/1.4047855 110804. [Online]. Available: https://doi.org/10.1115/1.4047855 [Page 6.]

[28] F. J. Montáns, F. Chinesta, R. Gómez-Bombarelli, and J. N. Kutz, "Data-driven modeling and learning in science and engineering," *Comptes Rendus Mécanique*, vol. 347, no. 11, pp. 845–855, 2019. [Page 8.]

[29] A. V. Roth, "Applications of empirical science in manufacturing and service operations," *Manufacturing & Service Operations Management*, vol. 9, no. 4, pp. 353–367, 2007. [Page 8.]

[30] U. Nations, "Transforming our world: the 2030 agenda for sustainable development," 2015. [Online]. Available: https://wedocs.unep.org/20.500.11822/9814 [Page 8.]

[31] P. Durana, N. Perkins, and K. Valaskova, "Artificial intelligence data-driven internet of things systems, real-time advanced analytics, and cyber-physical production networks in sustainable smart manufacturing," *Econ. Manag. Financ. Mark*, vol. 16, pp. 20–30, 2021. [Page 8.]

[32] C. Erazo, A. Yepes, S. Abolghasem, and G. Barbieri, "Mtconnect-based decision support system for local machine tool monitoring," *Procedia Computer Science*, vol. 180, pp. 69–78, 02 2021. doi: 10.1016/j.procs.2021.01.130 [Page 9.]

[33] Z. M. Çınar, A. Abdussalam Nuhu, Q. Zeeshan, O. Korhan, M. Asmael, and B. Safaei, "Machine learning in predictive maintenance towards sustainable smart manufacturing in industry 4.0," *Sustainability*, vol. 12, no. 19, p. 8211, 2020. [Pages 11 and 12.]

[34] R. Ahmad and S. Kamaruddin, "An overview of time-based and condition-based maintenance in industrial application," *Computers & industrial engineering*, vol. 63, no. 1, pp. 135–149, 2012. [Page 11.]

[35] P. J. Rivera Torres, E. I. Serrano Mercado, O. Llanes Santiago, and L. Anido Rifón, "Modeling preventive maintenance of manufacturing processes with probabilistic boolean networks with interventions," *Journal of Intelligent Manufacturing*, vol. 29, pp. 1941–1952, 2018. [Page 12.]

[36] A. Bousdekis, K. Lepenioti, D. Apostolou, and G. Mentzas, "Decision making in predictive maintenance: literature review and research agenda for industry 4.0," *IFAC-PapersOnLine*, vol. 52, no. 13, pp. 607–612, 2019. [Page 12.]

[37] M. Hermansa, M. Kozielski, M. Michalak, K. Szczyrba, Ł. Wróbel, and M. Sikora, "Sensor-based predictive maintenance with reduction of false alarms—a case study in heavy industry," *Sensors*, vol. 22, no. 1, p. 226, 2021. [Page 12.]

[38] T.-c. Fu, "A review on time series data mining," *Engineering Applications of Artificial Intelligence*, vol. 24, no. 1, pp. 164–181, 2011. [Page 12.]

[39] P. J. Brockwell and R. A. Davis, *Introduction to time series and forecasting*. Springer, 2002. [Page 13.]

[40] K. Shaukat, T. M. Alam, S. Luo, S. Shabbir, I. A. Hameed, J. Li, S. K. Abbas, and U. Javed, "A review of time-series anomaly detection techniques: A step to future perspectives," in *Advances in Information and Communication: Proceedings of the 2021 Future of Information and Communication Conference (FICC), Volume 1*. Springer, 2021, pp. 865–877. [Page 14.]

[41] Z.-H. Zhou, *Machine learning*. Springer Nature, 2021. [Pages 14 and 16.]

[42] M. W. Berry, A. Mohamed, and B. W. Yap, *Supervised and unsupervised learning for data science*. Springer, 2019. [Page 15.]

[43] A. Esteva, B. Kuprel, R. A. Novoa, J. Ko, S. M. Swetter, H. M. Blau, and S. Thrun, "Dermatologist-level classification of skin cancer with

deep neural networks," *nature*, vol. 542, no. 7639, pp. 115–118, 2017. [Page 16.]

[44] H. Wang, Z. Lei, X. Zhang, B. Zhou, and J. Peng, "A review of deep learning for renewable energy forecasting," *Energy Conversion and Management*, vol. 198, p. 111799, 2019. [Page 16.]

[45] Y. Ouali, C. Hudelot, and M. Tami, "An overview of deep semi-supervised learning," *arXiv preprint arXiv:2006.05278*, 2020. [Pages 16 and 17.]

[46] T. Ergen and S. S. Kozat, "Unsupervised anomaly detection with LSTM neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 8, pp. 3127–3141, aug 2020. doi: 10.1109/tnnls.2019.2935975. [Online]. Available: https://doi.org/10.1109%2Ftnnls.2019.2935975 [Page 17.]

[47] I. Guyon, S. Gunn, M. Nikravesh, and L. A. Zadeh, *Feature extraction: foundations and applications*. Springer, 2008, vol. 207. [Pages 17 and 18.]

[48] Y. Cui, P. Bangalore, and L. Bertling Tjernberg, "An anomaly detection approach using wavelet transform and artificial neural networks for condition monitoring of wind turbines' gearboxes," in *2018 Power Systems Computation Conference (PSCC)*, 2018. doi: 10.23919/PSCC.2018.8442916 pp. 1–7. [Page 18.]

[49] M. Ringnér, "What is principal component analysis?" *Nature biotechnology*, vol. 26, no. 3, pp. 303–304, 2008. [Page 18.]

[50] J. Cai, J. Luo, S. Wang, and S. Yang, "Feature selection in machine learning: A new perspective," *Neurocomputing*, vol. 300, pp. 70–79, 2018. [Page 18.]

[51] F. Giannoni, M. Mancini, and F. Marinelli, "Anomaly detection models for iot time series data," *arXiv preprint arXiv:1812.00890*, 2018. [Page 18.]

[52] X.-w. Chen and J. C. Jeong, "Enhanced recursive feature elimination," in *Sixth International Conference on Machine Learning and Applications (ICMLA 2007)*, 2007. doi: 10.1109/ICMLA.2007.35 pp. 429–435. [Page 18.]

[53] V. Fonti and E. Belitser, "Feature selection using lasso," *VU Amsterdam research paper in business analytics*, vol. 30, pp. 1–25, 2017. [Page 18.]

[54] A. Blázquez-García, A. Conde, U. Mori, and J. A. Lozano, "A review on outlier/anomaly detection in time series data," *ACM Comput. Surv.*, vol. 54, no. 3, apr 2021. doi: 10.1145/3444690. [Online]. Available: https://doi.org/10.1145/3444690 [Pages 19 and 20.]

[55] S. Basu and M. Meckesheimer, "Automatic outlier detection for time series: an application to sensor data," *Knowledge and Information Systems*, vol. 11, pp. 137–154, 2007. [Page 19.]

[56] K. M. Carter and W. W. Streilein, "Probabilistic reasoning for streaming anomaly detection," in *2012 IEEE Statistical Signal Processing Workshop (SSP)*, 2012. doi: 10.1109/SSP.2012.6319708 pp. 377–380. [Page 19.]

[57] J. Chen, W. Li, A. Lau, J. Cao, and K. Wang, "Automated load curve data cleansing in power systems," *IEEE Transactions on Smart Grid*, vol. 1, no. 2, pp. 213–221, 2010. doi: 10.1109/TSG.2010.2053052 [Page 19.]

[58] A. Reddy, M. Ordway-West, M. Lee, M. Dugan, J. Whitney, R. Kahana, B. Ford, J. Muedsam, A. Henslee, and M. Rao, "Using gaussian mixture models to detect outliers in seasonal univariate network traffic," in *2017 IEEE Security and Privacy Workshops (SPW)*, 2017. doi: 10.1109/SPW.2017.9 pp. 229–234. [Page 19.]

[59] J. Hochenbaum, O. S. Vallis, and A. Kejariwal, "Automatic anomaly detection in the cloud via statistical learning," *arXiv preprint arXiv:1704.07706*, 2017. [Page 19.]

[60] K. Hundman, V. Constantinou, C. Laporte, I. Colwell, and T. Soderstrom, "Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding," in *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, 2018, pp. 387–395. [Page 20.]

[61] V. Ishimtsev, I. Nazarov, A. Bernstein, and E. Burnaev, "Conformal k-nn anomaly detector for univariate data streams," 2017. [Page 20.]

[62] L. Duan, "Density-based clustering and anomaly detection," *Business intelligence-solution for business development*, pp. 79–96, 2012. [Page 20.]

[63] M. Khanam, "Afully online approach for anomaly detection and change-point detection in streaming data using lstms," Ph.D. dissertation, University of Manchester, 2022. [Page 20.]

[64] S. Schmidl, P. Wenig, and T. Papenbrock, "Anomaly detection in time series: a comprehensive evaluation," *Proceedings of the VLDB Endowment*, vol. 15, no. 9, pp. 1779–1797, 2022. [Pages 20 and 28.]

[65] Z. Ji, J. Gong, and J. Feng, "A novel deep learning approach for anomaly detection of time series data," *Scientific Programming*, vol. 2021, 2021. [Page 20.]

[66] D. Park, Y. Hoshi, and C. C. Kemp, "A multimodal anomaly detector for robot-assisted feeding using an lstm-based variational autoencoder," 2017. [Page 20.]

[67] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: Identifying density-based local outliers," *SIGMOD Rec.*, vol. 29, no. 2, p. 93–104, may 2000. doi: 10.1145/335191.335388. [Online]. Available: https://doi.org/10.1145/335191.335388 [Page 20.]

[68] Y. Gao, J. Lin, and C. Brif, "Ensemble grammar induction for detecting anomalies in time series," 2020. [Page 20.]

[69] Z. Li, Y. Zhao, N. Botta, C. Ionescu, and X. Hu, "COPOD: Copula-based outlier detection," in *2020 IEEE International Conference on Data Mining (ICDM)*. IEEE, nov 2020. doi: 10.1109/icdm50108.2020.00135. [Online]. Available: https://doi.org/10.1109%2Ficdm50108.2020.00135 [Page 20.]

[70] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *2008 Eighth IEEE International Conference on Data Mining*, 2008. doi: 10.1109/ICDM.2008.17 pp. 413–422. [Page 20.]

[71] O. Abdelrahman and P. Keikhosrokiani, "Assembly line anomaly detection and root cause analysis using machine learning," *IEEE Access*, vol. 8, pp. 189 661–189 672, 2020. doi: 10.1109/ACCESS.2020.3029826 [Page 21.]

[72] A. Saxena, K. Goebel, D. Simon, and N. Eklund, "Damage propagation modeling for aircraft engine run-to-failure simulation," in *2008 international conference on prognostics and health management*. IEEE, 2008, pp. 1–9. [Page 21.]

[73] G. Aydemir and B. Acar, "Anomaly monitoring improves remaining useful life estimation of industrial machinery," *Journal of Manufacturing Systems*, vol. 56, pp. 463–469, 2020. doi: https://doi.org/10.1016/j.jmsy.2020.06.014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0278612520301060 [Page 21.]

[74] L. Zhang, S. Elghazoly, and B. Tweedie, "Introducing anomdb: An unsupervised anomaly detection method for cnc machine control data," in *Annual Conference of the Prognostics and Health Management Society, Scottsdale, USA*, vol. 10, 2019. [Pages 21, 83, 86, and 87.]

[75] D. Y. Oh and I. D. Yun, "Residual error based anomaly detection using auto-encoder in smd machine sound," *Sensors*, vol. 18, no. 5, p. 1308, 2018. [Page 21.]

[76] G. S. Chadha, A. Rabbani, and A. Schwung, "Comparison of semi-supervised deep neural networks for anomaly detection in industrial processes," in *2019 IEEE 17th International Conference on Industrial Informatics (INDIN)*, vol. 1, 2019. doi: 10.1109/INDIN41052.2019.8972172 pp. 214–219. [Page 21.]

[77] J. Liu, K. Song, M. Feng, Y. Yan, Z. Tu, and L. Zhu, "Semi-supervised anomaly detection with dual prototypes autoencoder for industrial surface inspection," *Optics and Lasers in Engineering*, vol. 136, p. 106324, 2021. doi: https://doi.org/10.1016/j.optlaseng.2020.106324. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S014381662030498X [Page 21.]

[78] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, "Convolutional neural networks: an overview and application in radiology," *Insights into imaging*, vol. 9, pp. 611–629, 2018. [Pages 23 and 25.]

[79] M. W. Gardner and S. Dorling, "Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences," *Atmospheric environment*, vol. 32, no. 14-15, pp. 2627–2636, 1998. [Page 23.]

[80] W. Wang, Y. Yang, X. Wang, W. Wang, and J. Li, "Development of convolutional neural network and its application in image classification: a survey," *Optical Engineering*, vol. 58, no. 4, pp. 040 901–040 901, 2019. [Page 24.]

[81] L. Sadouk, "Cnn approaches for time series classification," *Time series analysis-data, methods, and applications*, vol. 5, 2019. [Page 25.]

[82] A. Shmilovici, "Support vector machines," *Data mining and knowledge discovery handbook*, pp. 231–247, 2010. [Pages 25, 26, and 27.]

[83] A. Mammone, M. Turchi, and N. Cristianini, "Support vector machines," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 1, no. 3, pp. 283–289, 2009. [Page 26.]

[84] M. Hejazi and Y. P. Singh, "One-class support vector machines approach to anomaly detection," *Applied Artificial Intelligence*, vol. 27, no. 5, pp. 351–366, 2013. doi: 10.1080/08839514.2013.785791. [Online]. Available: https://doi.org/10.1080/08839514.2013.785791 [Page 30.]

[85] W. Li, W. Hu, T. Chen, N. Chen, and C. Feng, "Stackvae-g: An efficient and interpretable model for time series anomaly detection," *arXiv preprint arXiv:2105.08397*, 2021. [Page 30.]

[86] M. A. Bashar and R. Nayak, "Tanogan: Time series anomaly detection with generative adversarial networks," in *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, 2020, pp. 1778–1785. [Page 30.]

[87] A. Dempster, F. Petitjean, and G. I. Webb, "ROCKET: exceptionally fast and accurate time series classification using random convolutional kernels," *Data Mining and Knowledge Discovery*, vol. 34, no. 5, pp. 1454–1495, jul 2020. doi: 10.1007/s10618-020-00701-z. [Online]. Available: https://doi.org/10.1007%2Fs10618-020-00701-z [Pages 32 and 33.]

[88] Z. Wang, W. Yan, and T. Oates, "Time series classification from scratch with deep neural networks: A strong baseline," 2016. [Page 32.]

[89] H. I. Fawaz, B. Lucas, G. Forestier, C. Pelletier, D. F. Schmidt, J. Weber, G. I. Webb, L. Idoumghar, P.-A. Muller, and F. Petitjean,

"InceptionTime: Finding AlexNet for time series classification," *Data Mining and Knowledge Discovery*, vol. 34, no. 6, pp. 1936–1962, sep 2020. doi: 10.1007/s10618-020-00710-y. [Online]. Available: https://doi.org/10.1007%2Fs10618-020-00710-y [Page 32.]

[90] J. Grabocka, N. Schilling, M. Wistuba, and L. Schmidt-Thieme, "Learning time-series shapelets," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '14. New York, NY, USA: Association for Computing Machinery, 2014. doi: 10.1145/2623330.2623613. ISBN 9781450329569 p. 392–401. [Online]. Available: https://doi.org/10.1145/2623330.2623613 [Page 33.]

[91] L. Beggel, B. X. Kausler, M. Schiegg, M. Pfeiffer, and B. Bischl, "Time series anomaly detection based on shapelet learning," *Computational Statistics*, vol. 34, pp. 945–976, 2019. [Page 33.]

[92] J. Dalzochio, R. Kunst, E. Pignaton, A. Binotto, S. Sanyal, J. Favilla, and J. Barbosa, "Machine learning and reasoning for predictive maintenance in industry 4.0: Current status and challenges," *Computers in Industry*, vol. 123, p. 103298, 2020. [Page 33.]

[93] H. Ali, M. N. M. Salleh, R. Saedudin, K. Hussain, and M. F. Mushtaq, "Imbalance class problems in data mining: A review," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 14, no. 3, pp. 1560–1571, 2019. [Page 33.]

[94] M. E. Villa-Pérez, M. A. Alvarez-Carmona, O. Loyola-Gonzalez, M. A. Medina-Pérez, J. C. Velazco-Rossell, and K.-K. R. Choo, "Semi-supervised anomaly detection algorithms: A comparative summary and future research directions," *Knowledge-Based Systems*, vol. 218, p. 106878, 2021. [Page 33.]

[95] E. Schubert and M. Gertz, "Intrinsic t-stochastic neighbor embedding for visualization and outlier detection: A remedy against the curse of dimensionality?" in *Similarity Search and Applications: 10th International Conference, SISAP 2017, Munich, Germany, October 4-6, 2017, Proceedings 10*. Springer, 2017, pp. 188–203. [Page 33.]

[96] N. Tatbul, T. J. Lee, S. Zdonik, M. Alam, and J. Gottschlich, "Precision and recall for time series," *Advances in neural information processing systems*, vol. 31, 2018. [Page 35.]

[97] R. F. Woolson, "Wilcoxon signed-rank test," *Wiley encyclopedia of clinical trials*, pp. 1–3, 2007. [Page 35.]

[98] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008. [Page 36.]

[99] J. M. Joyce, "Kullback-leibler divergence," in *International encyclopedia of statistical science*. Springer, 2011, pp. 720–722. [Page 36.]

[100] X. Gong and B. Lin, "Adding dummy variables: A simple approach for improved volatility forecasting in electricity market," *Journal of Management Science and Engineering*, vol. 8, no. 2, pp. 191–213, 2023. [Page 40.]

[101] J. A. Cortés-Ibáñez, S. González, J. J. Valle-Alonso, J. Luengo, S. García, and F. Herrera, "Preprocessing methodology for time series: An industrial world application case study," *Information Sciences*, vol. 514, pp. 385–401, 2020. [Page 41.]

[102] Z. Ding, G. Mei, S. Cuomo, Y. Li, and N. Xu, "Comparison of estimating missing values in iot time series data using different interpolation algorithms," *International Journal of Parallel Programming*, vol. 48, pp. 534–548, 2020. [Page 42.]

[103] J. Brownlee, "4 common machine learning data transforms for time series forecasting." [Page 43.]

[104] H. He, W. Tang, W. Wang, and P. Crits-Christoph, "Structural zeroes and zero-inflated models," *Shanghai archives of psychiatry*, vol. 26, pp. 236–42, 08 2014. doi: 10.3969/j.issn.1002-0829.2014.04.008 [Page 44.]

[105] H. Tang, Q. Wang, and G. Jiang, "Time series anomaly detection model based on multi-features," *Computational Intelligence and Neuroscience*, vol. 2022, 2022. [Page 44.]

[106] L. Kulanuwat, C. Chantrapornchai, M. Maleewong, P. Wongchaisuwat, S. Wimala, K. Sarinnapakorn, and S. Boonya-aroonnet, "Anomaly detection using a sliding window technique and data imputation with machine learning for hydrological time series," *Water*, vol. 13, no. 13, 2021. doi: 10.3390/w13131862. [Online]. Available: https://www.mdpi.com/2073-4441/13/13/1862 [Page 48.]

[107] M. Steininger, K. Kobs, P. Davidson, A. Krause, and A. Hotho, "Density-based weighting for imbalanced regression," *Machine Learning*, vol. 110, pp. 2187–2211, 2021. [Page 49.]

[108] A. N. Budynkov and S. I. Masolkin, "The problem of choosing the kernel for one-class support vector machines," *Automation and Remote Control*, vol. 78, pp. 138–145, 2017. [Page 50.]

[109] A. P. Bradley, "The use of the area under the roc curve in the evaluation of machine learning algorithms," *Pattern recognition*, vol. 30, no. 7, pp. 1145–1159, 1997. [Page 52.]

[110] T. Sakai, "Statistical significance, power, and sample sizes: A systematic review of sigir and tois, 2006-2015," in *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, 2016, pp. 5–14. [Page 52.]

[111] G. Aydemir and B. Acar, "Anomaly monitoring improves remaining useful life estimation of industrial machinery," *Journal of Manufacturing Systems*, vol. 56, pp. 463–469, 2020. [Page 92.]

TRITA-EECS-EX-2023:536