Transfer Learning for Predictive Maintenance Solutions

Doctoral Thesis

(Dissertation)

to be awarded the degree
Doctor of Engineering (Dr.-Ing.)

submitted by
Sebastian Schwendemann
from Lahr/Schwarzwald

approved by the Faculty of Mathematics/Computer Science
and Mechanical Engineering,
Clausthal University of Technology

Date of oral examination
December 20, 2023

Dissertation Clausthal, ISSE-Dissertation 27, 2023

Dean
Prof. Dr. Jörg P. Müller

Chairperson of the Board of Examiners
Prof. Dr. Rüdiger Ehlers

Chief Reviewer
Prof. Dr. Andreas Rausch

2. Reviewer
Prof. Dr.-Ing. Axel Sikora

3. Reviewer
Prof. Dr. Christian Siemers

D 104

Abstract

The last decades have seen the evolution of industrial production into more sophisticated processes. The development of specialized, high-end machines has increased the importance of predictive maintenance of mechanical systems to produce high-quality goods and avoid machine breakdowns. Predictive maintenance has two main objectives: to classify the current status of a machine component and to predict the maintenance interval by estimating its remaining useful life (RUL). Nowadays, both objectives are covered by machine learning and deep learning approaches and require large training datasets that are often not available. One possible solution may be transfer learning, where the knowledge of a larger dataset is transferred to a smaller one.

This thesis is primarily concerned with transfer learning for predictive maintenance for fault classification and RUL estimation. The first part presents the state-of-the-art machine learning techniques with a focus on techniques applicable to predictive maintenance tasks (Chapter 2). This is followed by a presentation of the machine tool background and current research that applies the previously explained machine learning techniques to predictive maintenance tasks (Chapter 3). One novelty of this thesis is that it introduces a new intermediate domain that represents data by focusing on the relevant information to allow the data to be used on different domains without losing relevant information (Chapter 4). The proposed solution is optimized for rotating elements. Therefore, the presented intermediate domain creates different layers by focusing on the fault frequencies of the rotating elements. Another novelty of this thesis is its semi and unsupervised transfer learning-based fault classification approach for different component types under different process conditions (Chapter 5). It is based on the intermediate domain utilized by a convolutional neural network (CNN). In addition, a novel unsupervised transfer learning loss function is presented based on the maximum mean discrepancy (MMD), one of the state-of-the-art algorithms. It extends the MMD by considering the intermediate domain layers; therefore, it is called layered maximum mean discrepancy (LMMD).

Another novelty is an RUL estimation transfer learning approach for different component types based on the data of accelerometers with low sampling rates (Chapter 6). It applies the feature extraction concepts of the classification approach: the presented intermediate domain and the convolutional layers. The features are then used as input for a long short-term memory (LSTM) network. The transfer learning is based on fixed feature extraction, where the trained convolutional layers are taken over. Only the LSTM network has to be trained again. The intermediate domain supports this transfer learning type, as it should be similar for different component types. In addition, it enables the practical usage of accelerometers with low sampling rates during transfer learning, which is an absolute novelty.

All presented novelties are validated in detailed case studies using the example of bearings (Chapter 7). In doing so, their superiority over state-of-the-art approaches is demonstrated.

Acknowledgments

The execution of a dissertation is a time-consuming task that takes place over a long period of time. During this period, some people have been particularly supportive of me. I would like to take this opportunity to thank them.

From a scientific point of view, I would like to thank my supervisor Prof. Dr. Andreas Rausch, who made it possible for me to pursue my doctoral degree at his institute. Further thanks go to my co-supervisor, Prof. Dr.-Ing. Axel Sikora for his scientific supervision during the entirety of my research work. Without his guidance, encouragement, and keen interest throughout our many hours of technical discussion, this thesis would not have been possible. Finally, I would also like to thank Zubair Amjad for his intensive cooperation with our project and the resulting impulses for my Ph.D.

Privately, I would like to thank my family for their understanding and the many hours they had to spend without me due to my dissertation. Special thanks go to my dear wife and my two sons. I would also like to thank my parents for their trust and support throughout my entire life. Unfortunately, my father will not be able to celebrate this graduation, as he passed away much too early, shortly before the completion of this thesis.

In addition, I would like to thank Erwin Junker Maschinenfabrik GmbH for giving me the opportunity to complete this Ph.D. and for providing the datasets. Finally, I want to give special thanks to my division manager, Mr. Johannes Schätzle, who made this possible for me in the first place.

Related Research

Parts of this thesis have been published in the form of research papers in the following journals:

- **S. Schwendemann**, Z. Amjad, and A. Sikora, “A survey of machine-learning techniques for condition monitoring and predictive maintenance of bearings in grinding machines,” *Computers in Industry*, vol. 125, p. 103380, 2021, doi: 10.1016/j.compind.2020.103380.
- **S. Schwendemann**, Z. Amjad, and A. Sikora, “Bearing fault diagnosis with intermediate domain based Layered Maximum Mean Discrepancy: A new transfer learning approach,” *Engineering Applications of Artificial Intelligence*, vol. 105, p. 104415, 2021, doi: 10.1016/j.engappai.2021.104415.
- **S. Schwendemann** and A. Sikora, “Transfer-Learning-Based Estimation of the Remaining Useful Life of Heterogeneous Bearing Types Using Low-Frequency Accelerometers,” *J. Imaging*, vol. 9, no. 2, p. 34, 2023, doi: 10.3390/jimaging9020034.
- **S. Schwendemann**, A. Rausch, and A. Sikora, “Detailed Study of Different Degradation Stages of Bearings in a Practical Reference Dataset,” in *2023 IEEE 28th International Conference on Emerging Technologies and Factory Automation (ETFA)*, Sinaia, Romania, 2023, doi: 10.1109/ETFA54631.2023.10275478.
- **S. Schwendemann**, A. Rausch, and A. Sikora, “A Hybrid Predictive Maintenance Solution for Fault Classification and Remaining Useful Life Estimation of Bearings Using Low-Cost Sensor Hardware,” 5th International Conference on Industry 4.0 and Smart Manufacturing, Lisbon, Portugal, 2023 [Status: accepted; not published yet]

The research results were also presented at the following conference:

- **S. Schwendemann,** “Predictive Maintenance: CNN basierende Zustandsanalyse von Kugellagern,” Internet of Things – vom Sensor bis zur Cloud 2020, Virtual conference (due to COVID-19), Oct. 21, 2020.
- **S. Schwendemann**, A. Rausch, and A. Sikora, “Detailed Study of Different Degradation Stages of Bearings in a Practical Reference Dataset,” in *2023 IEEE 28th International Conference on Emerging Technologies and Factory Automation (ETFA)*, Sinaia, Romania, 2023
- **S. Schwendemann**, A. Rausch, and A. Sikora, “A Hybrid Predictive Maintenance Solution for Fault Classification and Remaining Useful Life Estimation of Bearings Using Low-Cost Sensor Hardware,” 5th International Conference on Industry 4.0 and Smart Manufacturing, Lisbon, Portugal, 2023

Table of Contents

List of Figures

List of Tables

Abbreviations

AHHPMG	adaptive hybrid high-power multi-dimensional gradient
ANN	artificial neural network
Bi-LSTM	bidirectional long short-term memory
CNN	convolutional neural network
CNC	computerized numerical control
CORAL	correlation alignment
CSN	closed skew s-normal
CWRU	Case Western Reserve University
CWT	continuous wavelet transform
DA	domain adaptation
DAN	domain adaption network
DANN	domain adversarial neural network
DFT	discrete Fourier transform
DTN	deep transfer network
EMD	empirical mode decomposition
ERP	enterprise resource planning
FFT	fast Fourier transform
FTNN	feature-based transfer neural network
GAN	generative adversarial network
GRU	gated recurrent unit
HHT	Hilbert-Huang transform
HI	health indicator
I4.0	industry 4.0
IMF	intrinsic mode functions
IMS	Intelligent Maintenance Systems
JDA	joint distribution adaptation
JMMD	joint maximum mean discrepancy
LMMD	layered maximum mean discrepancy
LSTM	long short-term memory
MAE	mean absolute error
MAPE	mean absolute percentage error
MEMS	micro-electro-mechanical system

MK-MMD	multi kernel maximum mean discrepancy
ML	machine learning
MM	Markov model
MMD	maximum mean discrepancy
MSE	mean squared error
MTS	Mahalanobis Taguchi system
NRMSE	normalized root mean square error
PCA	principal component analysis
RCs	research challenges
ReLU	rectified linear unit
RQ	research question
RKHS	reproducing kernel Hilbert space
RMS	root mean square
RMSE	root mean squared error
RNN	recurrent neural network
RUL	remaining useful life
SMEs	small and medium-size enterprises
STFT	short-time Fourier transform
SVM	support vector machines
SVR	support vector regression
TK-MMD	tree kernel maximum mean discrepancy
TLA	transfer learning approach
WD-DTL	Wasserstein distance based deep transfer learning
WDCNN	wide deep convolutional neural network
WPE	wavelet packet energy

1 Introduction

1.1 Problem Statement

Current industrial production processes are becoming increasingly optimized and, thus, also increasingly complex. For this purpose, specialized high-end machines have been developed, which in turn lead to higher investment costs and, thus, higher machine hour costs [1]. Nowadays, these machines produce their workpieces through just-in-time production. If there are no disruptions in the supply chain, a workpiece is processed with nearly no further time buffer in the production chain. Therefore, it is desirable to keep the machine downtime as low as possible. Downtimes may be caused by maintenance intervals and unexpected failures. If one component of a machine fails, the entire line or even another company that depends on the produced workpiece may be forced to delay production. This is leading more and more companies to rely on predictive maintenance. In a recent Europe-wide poll of 1,550 industrial companies, 78% stated that they already use predictive maintenance mechanisms [2].

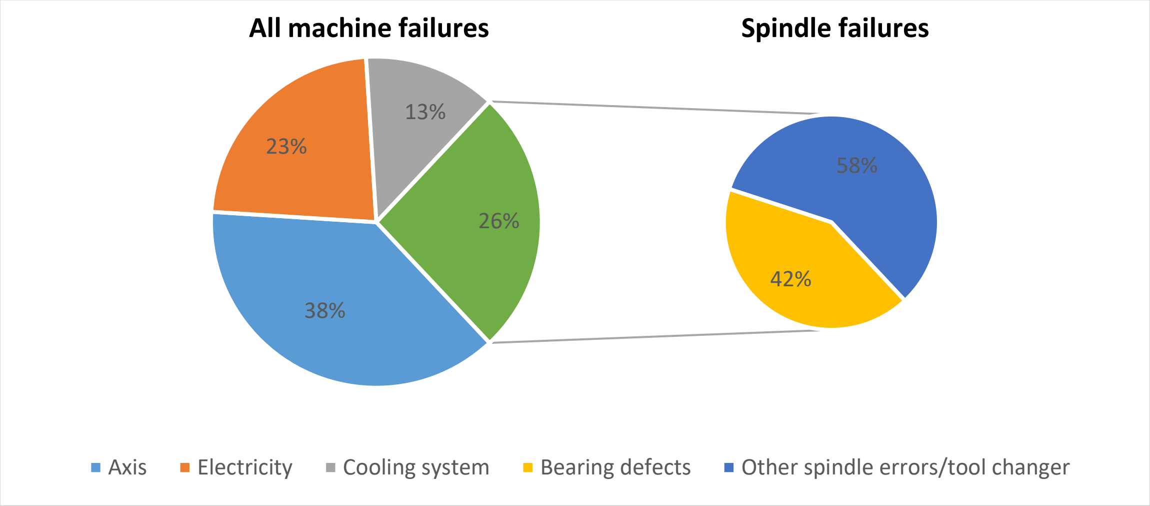

Figure 1: Failure probability on grinding machines. Twenty-six percent of all failures are based on spindle errors. Among those, 42% of the failures are due to bearing defects [3].

The machines monitored by a predictive maintenance process consist of many different components. Each of them is receptive to faults. The range of different fault types and their consequences becomes obvious when looking at the example of a grinding machine. A grinding machine is a machine tool that uses a grinding wheel made of hard material grains to produce a workpiece in a high-precision material removal process. Amongst components like axes or electrical components, one component that fails in 26% of their breakdowns is the grinding spindle or the tool changer [3]. Faulty bearings are the cause of grinding spindle failure in 42% of the cases [4] (see Figure 1). Changing a grinding spindle can take several hours. Therefore, to reduce the costs of a failure, it is essential that failures should be identified in advance. By detecting an impending failure, a replacement spindle can be ordered before a defect appears and maintenance can be planned.

Due to their complexity, complex defects, such as bearing defects, cannot be easily detected or predicted with traditional, non-machine learning-based methods like a simple threshold analysis [5]. Machine learning promises better solutions for problems that normally require a lot of manual fine-tuning or cannot even be solved at all using traditional techniques [6]. Therefore, one focus of this thesis is predictive maintenance using machine learning techniques. There are two areas of interest in the context of predictive maintenance: the detection of faults by means of classification and the estimation of remaining useful life (RUL).

An essential factor for machine learning is the availability of a large amount of data to train and test the parameters of the machine learning models. This is especially important for deep learning models, such as convolutional neural networks (CNN), which are based on "deep" layers of artificial neural networks and therefore have even more parameters to train. If the amount of data is too few, the trained model tends to overfit [6]. This means that the model matches for the samples in the training dataset but not for other samples. Most of the machine learning approaches need labeled data to train the model. This means that each datapoint should be assigned to a state that corresponds to the labels used for the classification results [7].

As stated above, there are a lot of specialized machines. Since all of them have their own setup with different components, there is no common data source that can be used for training a predictive maintenance system. Even on the same machine, the error pattern can vary based on process parameters like the rotational speed of a spindle. It also happens that measurement data is available but is not labeled completely. Disassembly of a pressed or welded component, such as an electronic device or a bearing, to determine the exact cause of a fault is demanding and time-consuming. The situation is made more difficult by the fact that many companies do not publish their data, and therefore this data cannot be used as a reference. For this reason, a lot of the current research is based on artificially created laboratory data [8]. In addition to the fact that the faults are often artificially created, there is another downside: the missing negative impact from other machine components in terms of background noise. However, there exist solutions to overcome the aforementioned lack of large, labeled datasets. One of these solutions is so-called transfer learning, which uses the knowledge of a large, labeled source dataset to improve the accuracy of a target dataset that can be small and unlabeled in the most unfavorable case [9]. Transfer learning can be used for error classification tasks as well as for remaining useful life tasks.

The conditions mentioned above make it nearly impossible for small companies to have a working predictive maintenance solution for their machines. Such companies require solutions that:

- Can handle the predictive maintenance task of real-world machines with a lot of noise.
- Can handle various process parameters like the rotational speed of the spindle.
- Can handle the problem of having only small datasets that are often not even labeled.
- Are based on machine learning techniques to avoid manual fine-tuning.
- Achieve better results through transfer learning.

1.2 Research Objectives

This thesis explores the research challenges around predictive maintenance solutions for use cases with a small amount of labeled, unlabeled, or rarely labeled data. The approaches presented in this thesis focus on transfer learning approaches to tackle the lack of data in a target domain. During the process of solving the research questions (RQs), the following research challenges (RCs) will be discussed.

RC1: Which methods are appropriate for predictive maintenance tasks of machines based on features of sensor data?

With the rise of machine learning over the last decade, a lot of research has been performed on the topic of predictive maintenance. Therefore, the current state of the art covers manifold methods. However, open questions are related to their usage for real life predictive maintenance scenarios.

- Are the methods appropriate for the analysis of sensor data?
- Which feature extraction methods are suitable for the needs of predictive maintenance?
- Which deep learning methods are available for this use case?

RC2: Under which constraints can the different methods be used?

This question is especially important for the different predictive maintenance scenarios that are also based on a variety of different dataset types. On the one hand, amongst others the following questions arise regarding feature extraction:

- Which methods are appropriate for stationary signals?
- Which are even usable with nonlinear and non-stationary signals?

On the other hand, different machine learning methods exist. Here, relevant questions include:

- Which machine learning methods are well suited for small training datasets.
- Which machine learning methods are only usable for large datasets?
- Which methods are suited for transfer learning to overcome the problem of small datasets?

RC3: How can existing methods be combined and optimized to complement each other?

There are many different methods, but it remains an open question how they can be used together to achieve optimal results. The combination of different feature extraction methods and transfer learning methods is of special interest here.

- Are data-driven feature extraction approaches like the Hilbert-Huang transform or hybrid approaches, such as handcrafted intermediate domains, better suited for this task?
- Based on the often-small datasets for a specific machine component, can transfer learning be a solution?
- Is it possible to use such a solution even for partly or unlabeled datasets?

These RCs are general and not specialized for a specific component. In contrast to this, this thesis also answers three bearing-specific research questions, which are defined and derived at the end of Chapter 3. These questions are the result of the presentation of the machine learning and predictive maintenance state of the art in Chapters 2 and 3.

The research questions are:

RQ1: What are the necessary characteristics of a new classification method, which can take benefits of a dataset of a different bearing type for a partly labeled target dataset that is collected under different process conditions?

RQ2: What are the necessary characteristics of a new RUL method, which can take benefits of a dataset of a different bearing type, for a labeled target dataset that is recorded with sensors with low sampling rates?

RQ3: What are the necessary characteristics of a feature extraction method that is well suited for transfer learning? This method must be stable enough to be used on different bearing types without changing its parametrization or making significant changes to a subsequent machine learning model for different bearing types.

In order to answer these three RQs, it is essential to know which methods are appropriate for predictive maintenance tasks for machines based on sensor data and under which constraints they can be used. In addition, the answers may appropriately combine these methods. This is what the RCs are about. Therefore, the findings of the RCs are used as input for answering the RQs. As can be seen in Table 1, this results in the RCs being answered at the beginning of this thesis. The answers for the RQs that build on the answers for the RCs are given afterward.

In order to be able to assign the content of the individual chapters to the questions, a brief assignment of their content to the questions is made at the end of each chapter. In addition, a detailed answer for each question is given in Chapter 8 of this thesis.

Table 1: Organization of the research questions in the chapters of this thesis.

Research Question	Chapter					
	2	3	4	5	6	7
RC1: Methods	🟦	🟦				🟦
RC2: Constraints	🟦	🟦				🟦
RC3: Combinations & Optimizations			🟦	🟦	🟦	🟦
RQ1: Classification				🟦		🟦
RQ2: Remaining Useful Life					🟦	🟦
RQ3: Intermediate Domain			🟦			🟦

By answering the research questions, the three main contributions of this thesis are developed:

1. A new layer-based intermediate domain that focuses on fault frequencies of bearings. This intermediate domain can be used on different bearing types and process conditions without any modifications. This intermediate domain is well suited for transfer learning and is the answer to RQ3, which asks for such a feature extraction mechanism.
2. The first transfer learning approach for fault classification of bearings that supports a knowledge transfer between different component types and, at the same time, significantly different process parameters. For this purpose, a CNN solution based on the new intermediate domain is presented. In addition, a new loss function that extends an existing loss function called maximum mean discrepancy (MMD) is presented. This new loss function for training a neural network uses the characteristics of the intermediate domain and is called layered MMD (LMMD). This approach directly targets RQ1, which asks for a new classification approach for fault classification of partly labeled datasets.
3. The first transfer learning approach for RUL estimation between different types that is based on data from accelerometers with low sampling rates. Here, a feature extraction mechanism based on the intermediate domain and convolutional layers is used in combination with a long short-term memory (LSTM) network. This approach directly targets RQ2, which asks for a new RUL estimation method for labeled datasets.

All novelties are presented and evaluated based on the concrete example of bearings but might be adaptable to other components with periodic movements, such as gears.

1.3 Thesis Outline

Building on the previous chapter, which assigned the research questions to the different chapters of this thesis, this chapter will explain the individual chapters and their contents. Chapter 2 of this thesis introduces a comprehensive background of the machine learning concept in general as well as for the specific task of transfer learning for predictive maintenance, which covers, for instance, convolutional neural networks.

Chapter 3 illustrates the topic of predictive maintenance. This is done by ranking predictive maintenance in the context of Industry 4.0. Afterward, an introduction to the background of grinding machines, spindles, and bearings is given. In addition, the chapter presents the current research on predictive maintenance with a focus on the state of the art for transfer learning in the context of bearings in grinding spindles. This includes CNN-based transfer learning for bearing fault classification and recurrent neural network (RNN)/LSTM-based transfer learning for estimating the remaining useful life of bearings. Finally, the three research questions are defined based on the previously presented state of the art.

In Chapter 4, a novel feature extraction method based on an intermediate domain that focuses on relevant information of processes with rotating elements is presented. This intermediate domain comes without a loss of relevant information.

A novel solution for transfer learning-based fault classification is presented in Chapter 5. This covers a CNN-based transfer learning architecture that uses the in Chapter 4 introduced intermediate domain as input. In addition, a new domain adaptation loss function called LMMD is introduced.

The next chapter (Chapter 6) presents a solution for the transfer learning task of the RUL scenario. Therefore, the feature extraction part of the classification task is reused. This reused part is based on the intermediate domain of Chapter 4 and the convolutional layers of Chapter 5. An LSTM network is used for the RUL estimation to cover the time dependencies. In addition, a transfer learning approach based on fixed feature extraction of the convolutional layers is presented. This approach also has a particular focus on being usable for accelerometers with low sampling rates.

In Chapter 7, both presented solutions are demonstrated and verified in case studies. For the classification part, this is done by an exploration based on three reference datasets. Two of them are bearing datasets available in the public domain, and one is a private one of bearing faults in grinding spindles. In unsupervised and semi-supervised scenarios, other feature extraction methods and loss functions are compared to the intermediate domain and LMMD. This is followed by a benchmark against a current state-of-the-art approach. Finally, a verification of the RUL approach is given based on the execution of the IEEE PHM Data Challenge 2012, which was a challenge for the estimation of the RUL for bearings.

The last chapter (Chapter 8) summarizes the main contributions and conclusions of this thesis, which answer the research questions and suggest recommendations for future work.

2 Machine Learning Background

2.1 Introduction

This chapter presents the background and current state of the art for machine learning (ML) and deep learning in the context of predictive maintenance solutions. First, machine learning in general is introduced in Section 2.2. Section 2.3 describes important feature extraction methods for time data. Section 2.4 illustrates the theoretical backgrounds of transfer learning and describes the state of the art for transfer learning for deep learning techniques used later in this thesis.

2.2 Machine Learning in General

2.2.1 Introduction

According to Arthur Samuel, machine learning is the “field of study that gives computers the ability to learn without being explicitly programmed” [10]. This involves the following benefits of machine learning compared to traditional programming techniques [6]:

- It can perform better and simplify programming for problems where a lot of hand-tuning is required, or long lists of rules are needed.
- It can provide solutions for complex use cases where no solution can be found with classic algorithms, for instance, very large datasets.
- It can automatically adapt to new or dynamic datasets.

These benefits, combined with the current trend to collect more and more data, which results in large available training datasets [6], are leading to an increased usage of ML approaches [11].

An ML process consists of two parts [6]: The training data that can be used directly or in the form of extracted features and the machine learning model. This section introduces the theoretical backgrounds of machine learning, such as features and the relevant machine learning models for predictive maintenance tasks.

2.2.2 Theoretical Background

There are different types of machine learning algorithms whose suitability for a particular task can be made by classifying them into different categories. The most relevant categories are illustrated in Figure 2 and further described in the following section.

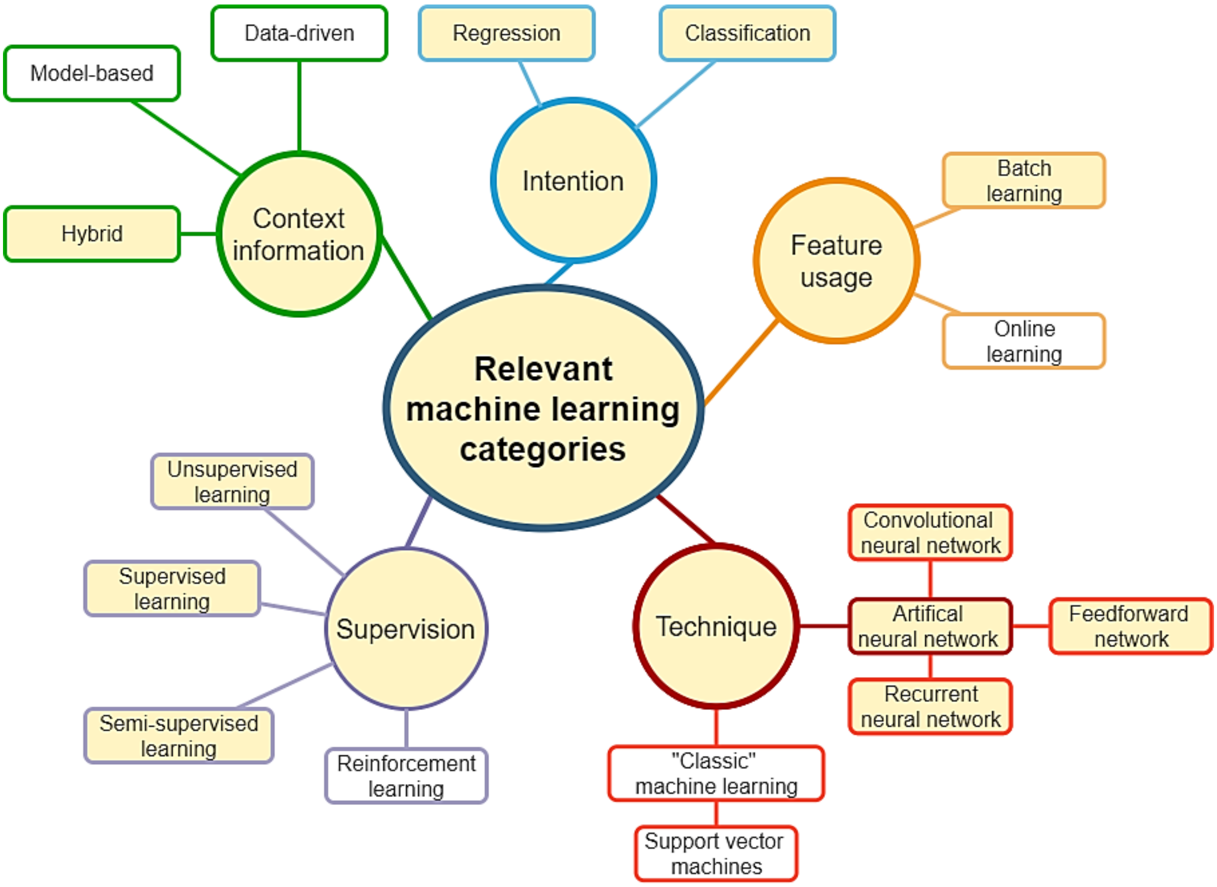

Figure 2: Relevant machine learning categories in general and the ones directly used in this thesis (highlighted). All machine learning processes can be assigned to a subitem of each of the five main categories.

One possibility to categorize machine learning is by its usage of the available context information. Often, machine learning approaches are purely data-driven and rely on the effective extraction of features of the data that are used as input for the machine learning network [12]. In contrast to them, there are the so-called model-based approaches. They build a physical model to describe the current state of a machine. This is often done by means of equations (algebraic or differential). Therefore, knowledge of the system and its failure mechanism is needed. However, this may not be applicable to complex systems due to the lack of a complete understanding of the whole system [13]. Nevertheless, some approaches combine model-based and data-driven approaches. These so-called hybrid approaches close the gap between both by using the model-based domain knowledge and the data-driven machine learning algorithm [13, 14]. Those solutions tend to be more accurate because of the combination of the advantages of both approaches [14].

Another type of categorization is by the intention of the given problem. This can be a classification or a regression problem [6]. In a classification problem, the algorithm selects a label for a given input. For example, there may be only two labels in an easy predictive maintenance setup: healthy and defect. However, regression predicts a continuous quantity, like sensor values or the remaining useful life of a component. The algorithms for these categories do not have to be limited to one kind of problem. Some algorithms, such as support vector machines (SVM), can be used for both.

In addition, machine learning techniques can also be categorized based on the amount of supervision required during the training [6]:

- Supervised learning: The data must be labeled for the training algorithm for using supervised learning. This means that each data sample must be assigned to a class. The algorithm utilizes the input and the labels to learn a mapping between them. Afterward, this mapping is applied to unseen data. Supervised learning is the most-used method since it can deliver the best results. The downside is that it is not always possible since there are scenarios where a lack of labels in the training data exists.
- Unsupervised learning: These algorithms do not use labeled datasets for unsupervised learning. Their principle is to learn inherent latent structures and relationships from the input data. Usually, they are used for clustering (detecting similar groups), dimension reduction (reducing the number of features), anomaly detection (detecting samples that are different from the ordinary ones), and transfer learning (transferring knowledge from one dataset to another).
- Semi-supervised learning: This is a combination of supervised and unsupervised learning, where a small amount of labeled data and a large amount of unlabeled data are usually available.
- Reinforcement learning: This approach is based on an agent who trains incrementally. Therefore, it rates the results of the current iteration by giving a reward. The algorithm tries to maximize the reward by changing its strategy. Since reinforcement learning, incremental learning, and online learning are all closely related and are not the focus of this thesis, they will only be referred to as incremental learning from now on. For more details about this topic, see Taylor and Stone [15].

A further criterion for differentiation between machine learning algorithms is how the features are used for the training. The training can be done with batch learning but also with online learning. Batch learning, also known as offline learning, is a learning type where a model can only be trained with the features of all samples of a dataset together. When new data appears, the model must be retrained completely. In contrast, only the features of a new sample are used to optimize the existing trained model when learning online. In addition, the samples can also be accumulated in small groups—the so-called mini-batches [6, 16].

There are different machine learning techniques available that can be assigned to the above-mentioned categories. One well-known technique is an SVM that uses human-hand-crafted features of the input data and transforms them into a higher dimensional space representation to solve ML tasks more easily. The downside of those algorithms is that they are hard to scale to very large datasets and are not very successful for voice recognition and image classification tasks [17]. A subfield of ML is deep learning. In contrast to techniques like SVM, deep learning is based on artificial neural networks (ANN) that are arranged in "deep" stacked layers. Therefore, they are also called deep neural networks. The benefit of these networks is that they can extract the best features automatically [6].

There are different variants of ANNs that are used in deep neural networks. However, the specializations convolutional neural networks (CNN) and recurrent neural networks (RNN) are the most important ANNs [18]. Therefore, in addition to ANNs in general, these two deep-learning networks will be discussed in the next three sections. One of the most important traditional machine learning models, the support vector machine, will also be explained in brief [6].

2.2.3 Artificial Neural Networks

An artificial neural network (ANN) is a generic term for all computing systems that are inspired by biological neural networks [6]. An ANN consists of interconnected, artificial neurons, mimicking neurons in a human brain. When an artificial neuron receives an input signal, it processes it and then forwards the output signal to all other directly connected neurons.

The output signal of an artificial neuron is calculated with the help of three elements: the weighted sum of the inputs, a bias, which on a trained neuron is a static offset, and an activation function. Often, the output must reach a threshold until the signal is sent to its connected neuron.

An ANN consists of neurons that are commonly grouped in at least three layers. The input layer accepts the inputs, and the output layer delivers the results of the entire ANN. In addition, there can be several hidden layers in-between these layers, in which the states are not accessible from the outside. The most-used type of network is a feedforward network, which has one data direction only [6]: the output of each layer is the input of the next layer. Layers can be fully connected, where every output of the previous layer connects with a certain weight to every node of the following layer. Before an ANN can be used, it must be trained. The training can be done by backpropagation, wherein the neural network's output is compared with the expected result. The difference is returned as an error to the previous layers. Subsequently, they readjust their weights according to the error.

2.2.4 Convolutional Neural Network

Convolutional neural networks (CNN) are a specialized type of ANN. They were first designed and proposed by LeCun et al. [19] and later optimized using an error-gradient algorithm. Since CNNs have the unique ability to maintain initial information regardless of shift, scale, and distortion invariance, which is basically achieved through local receptive fields, they are widely used in image classification, traffic sign recognition, and many other applications [20–25].

CNNs are inspired by the brain's visual cortex, which consists of many neurons, each with a small receptive field. This means that they only react to a small region of interest. Furthermore, these fields may overlap [26]. The architecture of a CNN is often similar, consisting of a stack of different layers of neurons. Typically, a CNN has the following structure: convolutional layers followed by a pooling layer, followed by additional convolutional layers and a pooling layer, and so on. At the end, there is a regular feed-forward neural network with a few fully connected layers (see Figure 3) [6]. Such a fully connected layer is also called a dense layer.

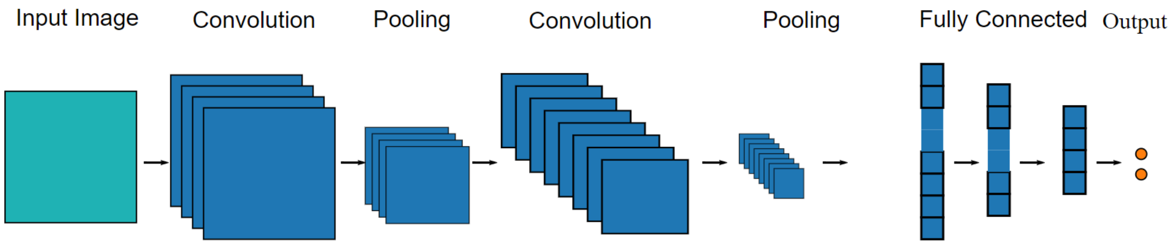

Figure 3: Typical CNN architecture. The input image is first processed with convolutional and pooling layers, followed later by fully connected layers.

A convolutional layer does not use the output of the previous layer as a whole. Each neuron has a small field of view only (also known as a window or kernel). For each pixel in its view, its value is combined with a filter, and its weight is used to generate one output pixel.

A pooling layer has a similar concept as a convolutional layer. It takes data from a window of the previous layer and calculates a new pixel. The difference between a pooling layer and a convolutional layer is that a pooling layer aims to reduce the size of an image by removing unnecessary information/pixels. The neurons in a pooling layer do not have a weight. They use a simple aggregation function (e.g., max or mean) on the data of the input window.

As shown in Figure 3, the output of the last fully connected layer is used for the classification. The specific layout of CNNs leads to their strength in image classification. The lower layers are used for the more generic low-level feature extraction, while the highest layers are used for the classification itself [27].

In addition to the common use case for classifying 2D images, CNNs can also be used for 1D data of time series, such as sensor data. The space between the cells and the filters used helps the neural network to interpret the different observations in the time series data [28].

2.2.5 Recurrent Neural Network

A recurrent neural network (RNN) is an ANN that analyzes time series to predict their future. This is in contrast to CNNs, which are more suitable for classification tasks. RNNs' time series capabilities are based on their ability to remember input data of previous time steps and use them as additional input for the current time step [29]. This makes them suitable for predictive maintenance solutions that rely on time series data, like predicting the remaining useful life of a machine [30].

The information flow of an RNN is as follows: A recurrent neuron has the output vector of the last time step as additional input to its input vector. As shown in Figure 4, the output y_{t-2} of the neuron on time position *t-2* is used for two tasks. First, it is an output like in other neural networks, but it is also an additional input for the next time step (*t-1*). The output y_{t-1} is again an additional input of time *t*. With this mechanism, the results respect the results of the previous iteration as well as the new input. Each neuron has two weights: one for the input and one for the output of the last time step. Usually, RNNs are used as deep RNNs. Therefore, they are stacked in layers of cells. A deep RNN can have, for instance, three layers and 100 neurons in each layer [6].

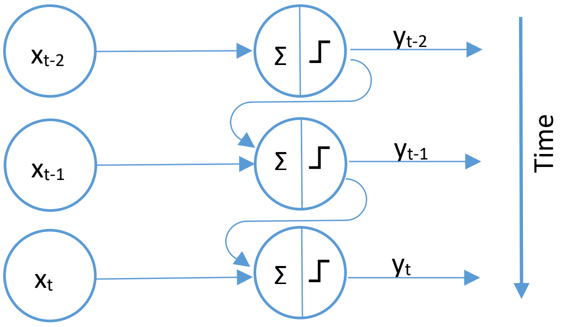

Figure 4: Information flow of an RNN through time.

The downside of RNNs is that they can suffer from the vanishing gradient problem, which can occur during the training wherein the gradient is used to update the weights through backpropagation [6]. The gradient may diminish through time because of the backpropagation. As a result, the network does not learn correctly and has a short-term memory. Two new network types have been developed to overcome this problem: Long short-term memory (LSTM) and gated recurrent unit (GRU). Both can learn long-term dependencies using a mechanism called gates. These gates learn which information of a sequence is important and which could be dropped. LSTMs have three gates: (input (i_t), forget (f_t), and output (o_t)) and the cell state (C_t) (see Figure 5). An LSTM can learn which input is important (input gate) and store it in a log-term state. With the help of the forget gate, an LSTM learns how long to store the input. A GRU cell works quite similarly by replacing the cell state with a hidden state to transfer information (Figure 6). A GRU consists of the reset (r_t) and update (z_t) gates and the hidden state (h_t).

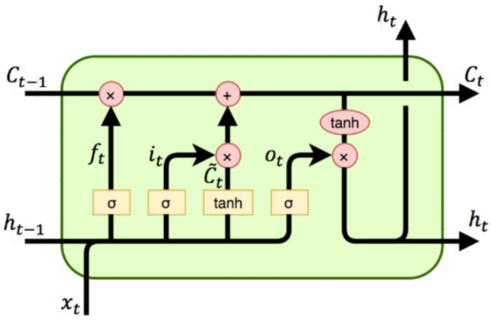

Figure 5: Architecture of an LSTM cell with the three gates: input (i_t), forget (f_t), and output (o_t) [31].

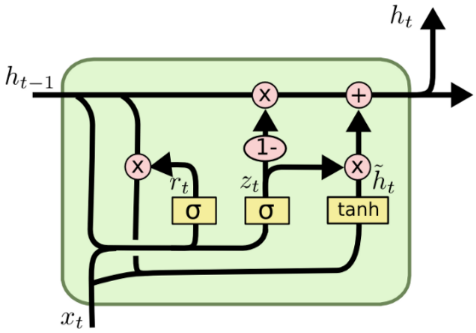

Figure 6: Architecture of a GRU cell with its two gates: reset (r_t), update (z_t); and the hidden state (h_t) [31].

2.2.6 Support Vector Machine

A support vector machine (SVM) is a simple but effective mechanism for classification [6]. An SVM tries to separate the values of the features in separate areas with a linear separator (see Figure 7). Parallel to this line, two (here dashed) lines represent the decision boundary, while the area in between is called the hyperplane. Each dot is called a support vector. Classification is done by checking which side of the decision boundary the instance is on.

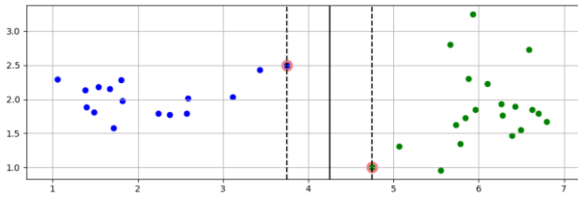

Figure 7: A visualization of an SVM classification. The red dots are the support vectors, and the dashed lines define the decision boundaries of the green and blue classes.

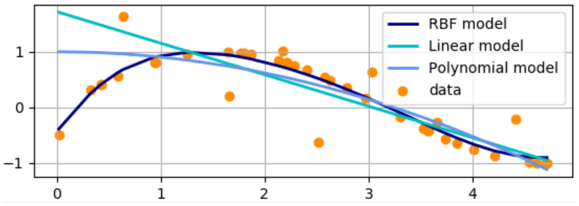

Figure 8: An SVR with different kernel models to create the hyperplane for a regression task.

In contrast to an SVM, support vector regression (SVR) is a machine learning algorithm for regression tasks. The architecture of an SVR and an SVM is quite similar. The difference is that an SVR tries to have as many training instances as possible inside the hyperplane, which is used to estimate the values during the regression task.

For nonlinear problems, the so-called kernel trick must be performed. As shown in Figure 8, the fields can be separated by polynomial or Gaussian RBF lines instead of a linear line [6].

2.3 Feature Extraction Methods for Predictive Maintenance

2.3.1 Introduction

Before any of the above-presented machine learning models can be used for a predictive maintenance task, feature extraction methods should be used to preprocess the data. In many scenarios, sensor data recorded over a certain period of time is used. In addition, this data is often noisy due to influences from other components. Feature extraction has the following aims [6, 32]:

- Reducing the amount of data. This is especially important for time series such as sensor data.

- Combining existing features to create a new, more powerful feature. For instance, in the case of sensors, this can be time and measured values.
- Decreasing the computational costs based on unused features.
- Decreasing the probability of overfitting based on unused features.

All feature extraction techniques within the scope of time-dependent sensor data can be categorized into three categories (see Figure 9): time domain, frequency domain, and time-frequency domain. In the following sections, these techniques will be explained in detail.

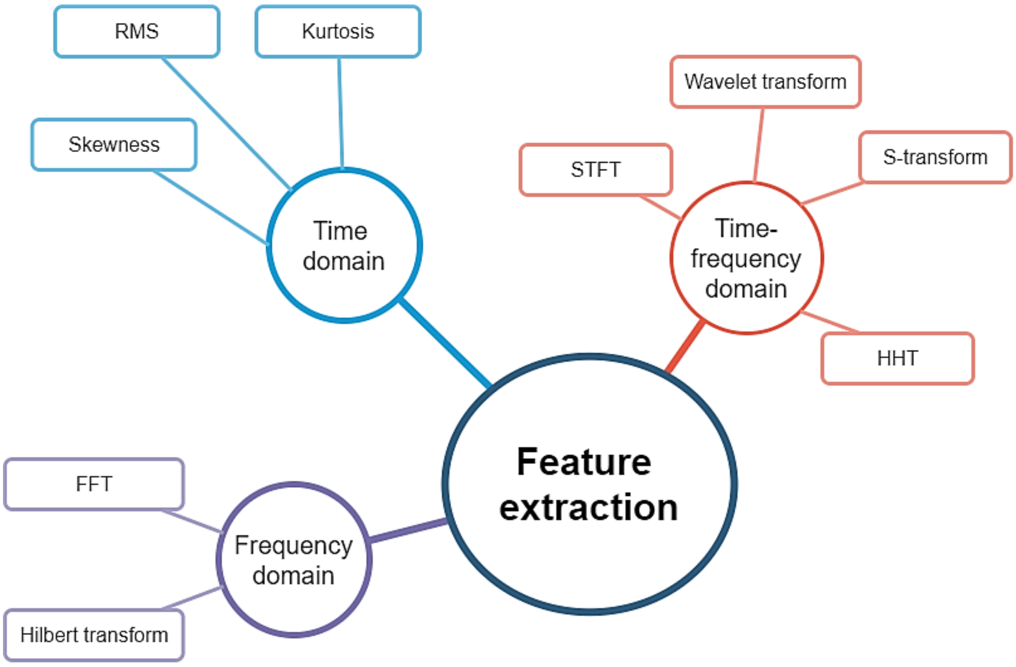

Figure 9: Feature extraction techniques for time signals. There are techniques in the time domain, the frequency domain, and the time-frequency domain.

2.3.2 Time Domain Analysis

A time domain analysis uses the direct raw sensor data. There exist approaches based on descriptive statistics like mean, peak-to-peak, amplitude, and standard deviation. However, high-order statistics like root mean square (RMS) and kurtosis are also common [33, 34].

2.3.3 Frequency Domain Analysis

The big advantage of the frequency domain over the time domain is its ability to focus on frequency components of interest, like specific fault frequencies. The first step of frequency domain analysis is to transform the data into the frequency domain. This is often done via a Fourier transform [35]. Power spectral density is another common technique in the frequency domain [36]. Frequency filters and envelope analysis are also of interest [37].

2.3.3.1 Fast Fourier Transform

With the help of a discrete Fourier transform (DFT), it is possible to transform signals from time to frequency domain. Afterward, the amplitude of each frequency can be analyzed. A DFT is defined by Eq. (1), where y_i is a sample of the time domain with a total size of N samples. $Y_k$is the transformed discrete Fourier coefficient, $k \in [0; N-1]$[35].

$$Y_k = \sum_{n=0}^{\mathrm{N}-1} y_i * e^{-\frac{i\,2\pi\,k\,n}{N}} \tag{1}$$

Since the calculation of the DFT is rather complex ($O(n^2)$), the fast Fourier transform (FFT) is usually used for transformations into the frequency domain. An FFT has only a complexity of $O(n * \log n)$ [38]. The first step when using an FFT is to choose the time window to be analyzed. This can be the whole data series or a subseries. Figure 10 shows an example of a transformation of a whole signal based on two overlaying frequencies (55 Hz and 100 Hz) to the frequency domain with the help of an FFT.

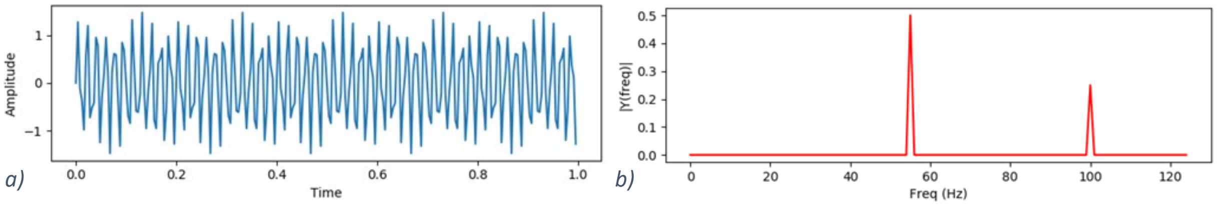

Figure 10: Transformation of a time signal (a) to the frequency domain (b) with the help of an FFT. This signal consists of two frequency signals: One at 55 Hz and the other at 100 Hz.

2.3.3.2 Envelope Analysis

The envelope analysis converts time data into the envelope spectrum. It is a widely used method to analyze data from mechanical systems by extracting periodic signals from modulated random noise [37].

The resonance frequency of a mechanical system is generated by the amplitude modulation and the carrier frequency of a vibrations signal. With the help of the envelope analysis, those two signals can be separated [39]. Two steps are used to calculate the envelope spectrum. First, the envelope is calculated by applying the Hilbert transform to the time domain signal. This transformation calculates an analytic signal, which is a complex-valued function that has no negative frequency components. The analytic signal that is used for the recovery of the modulation signal (demodulation) is calculated by Eq.(2)

$$H^{\{x(t)\}} = \frac{1}{\pi}\mathrm{p.v.}\int_{-\infty}^{+\infty} \frac{x(\tau)}{t-\tau} d\tau \tag{2}$$

where p.v. is the Cauchy principal value of the integral process and *x(t)* is a simple periodic signal according to Eq. (3) [39]. *A* is the amplitude of the analytic signal, and *f* is its frequency.

$$x(t) = \mathrm{A} * \cos{(2\pi\mathrm{ft})} \tag{3}$$

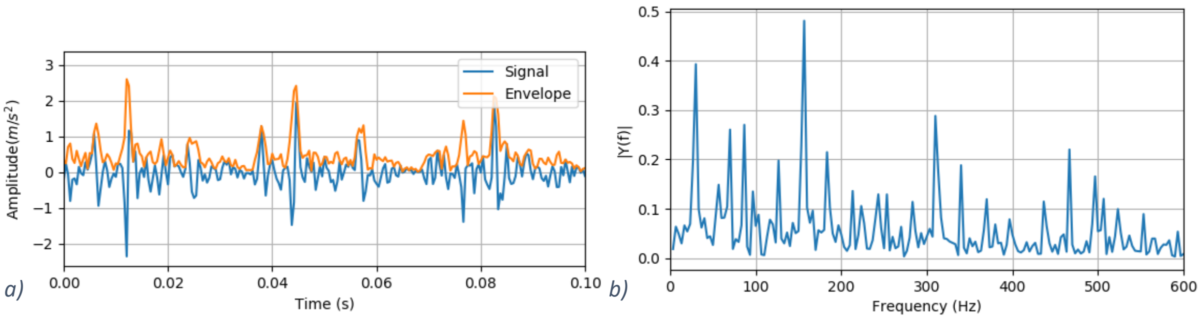

Figure 11: This figure shows the envelope (a) calculated for a vibration signal, which is further processed to get the envelope spectra (b). Here the frequency of 156 Hz and its harmonics (312 Hz, 480 Hz, and so on) are well recognizable.

As shown in Figure 11, the envelope, which is the result of the Hilbert transform, is later used to extract the periodically occurring frequencies inclusive of its harmonics by applying an FFT.

2.3.4 Time-Frequency Analysis

2.3.4.1 Introduction

A big disadvantage of the frequency domain analysis is that all time-related information is lost after applying the transformation into the frequency domain [40]. This leads to the delivery of a time-averaged spectrum, which prevents nonlinear and non-stationary signals from being analyzed correctly [41]. A nonlinear signal is a signal of a nonlinear dynamic process, which implies that it is a partial solution of a nonlinear differential equation. The signal can also be non-stationary, meaning it lacks a specific mapping rule, and the first, second, or higher moments of the underlying stochastic processes vary over time [42, 43]. Covering nonlinear and non-stationary signals is important because real-world systems, such as machines, are often based on them [43].

To analyze these signals, several techniques combine the information from the time domain and the frequency domain. These might be extensions to existing frequency domain analyses like a short-time Fourier transform (STFT). However, there are also time-frequency specific techniques such as the continuous wavelet transform (CWT), the Hilbert-Huang transform (HHT) in combination with the empirical mode decomposition (EMD), and the Stockwell transform (S-transform). In the following section, these methods are explained in detail.

2.3.4.2 Short-Time Fourier Transform

The STFT is an extension of the FFT. An STFT removes the loss of time information, which happens during an FFT by splitting the data series into small equal-sized sequences, also called windows. For each window, a separate FFT is calculated. As a result, there exists a mapping between time slices with the size of the window and their corresponding frequencies. The amplitude of the frequencies is not calculated for a single point in time. Instead, it is calculated for the whole window. Figure 12a shows an increasing linear signal of a 1-second length. The window size of the STFT is $0.\overline{3}$ seconds. This example shows different amplitudes for the frequency in each window (see Figure 12b). If the resolution has to be higher, the window size has to be decreased. It is also possible to use a sliding window. Here the windows are not consecutive but slightly overlap each other. With the help of the sliding window, more windows can be generated out of the same signal length [44].

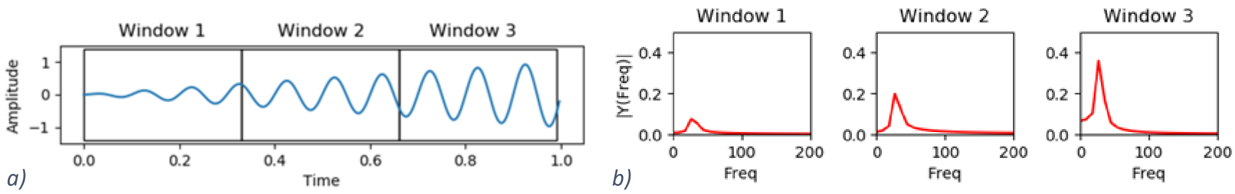

Figure 12: STFT of a 1 s signal with a 0.$\overline{3}$ s window size: a) shows three windows used for the STFT in the time domain; b) shows the resulting FFTs of each window.

STFTs are easy to apply since FFTs are a well-known technique. The downside of this approach is that there is no continuous frequency resolution. The resolution depends on the window size. Smaller windows lead to a good time resolution, but the resulting frequency domain is less accurate than that of larger windows. This is important when analyzing data that have components in both a high-frequency and low-frequency range. For low frequencies, the aim is to achieve a good (absolute) frequency resolution since a small absolute frequency change is especially important here. By contrast, a good time resolution is important at high frequencies since a complete oscillation takes less time, and, therefore, the instantaneous frequency can change more quickly [45].

The windowing can also lead to another disadvantage of the STFT: the so-called leakage effect [46]. For an exact representation, a window that matches the period length of the time series frequencies is needed. However, this is not always possible, especially when different frequencies are of interest. As a result of the non-matching window size, additional frequencies that do not exist in the signal itself can appear. These additional frequencies take some power from the frequencies representing the signal. Therefore, the original frequencies have a lower amplitude.

2.3.4.3 Continuous Wavelet Transform

Wavelet transformations overcome the aforementioned limitation of an STFT being only usable in a specific frequency range as well as its vulnerability for the leakage effect [46]. Both algorithms are similar in that they disassemble signals from the time domain into an infinite amount of substitute functions. The STFT uses sinusoidal functions. Wavelet transformations replace these sinusoidal functions with wavelet functions. Wavelets are oscillatory functions with a limited duration.

The original wavelets are called “mother” wavelets. During a CWT, the mother wavelet has to be scaled and translated so that they match the signal. During the scaling, the energy of a wavelet is always the same as that of the mother wavelet. It is shrunk into one direction (e.g., x-axis) and stretched into the other direction (see Figure 13).

A CWT is expressed by the following integral [46]:

$$W(s,\tau) = \frac{1}{\sqrt{s}} \int_{-\infty}^{+\infty} x(t)\psi * \left(\frac{t-\tau}{s}\right) dt, \tag{4}$$

Where s is the scale and τ is the translation (time) parameter. ψ is the chosen mother wavelet. During a CWT, n scales must be used. Starting at scale 1, the wavelet must be moved (translation) to each data point. After that, it must be measured how well it fits the input curve. This algorithm must be performed for the entire time series. The result is a coefficient for each point. Afterward, the same procedure must be performed with the next scales from 2 to n.

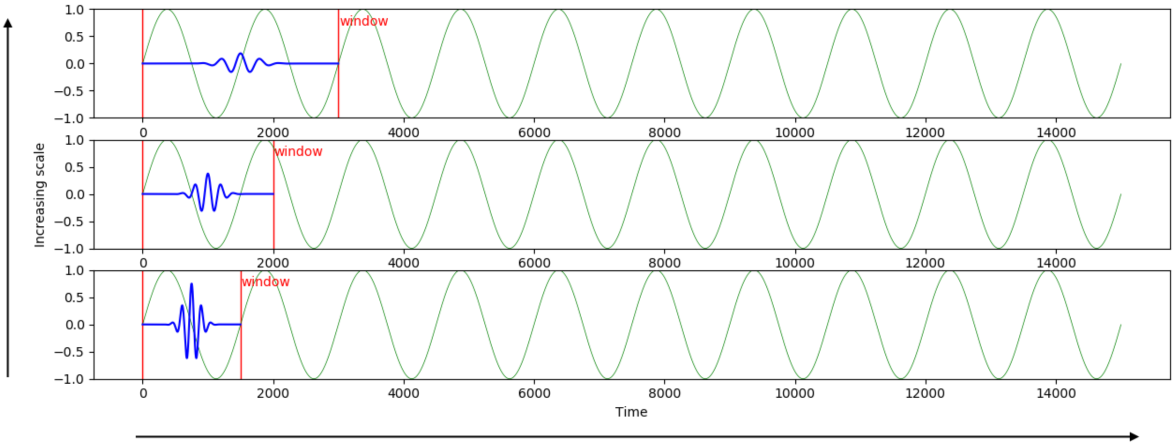

Figure 13: Different scales of a Morlet wavelet. An increased scale leads to a wider window in time that a wavelet can cover.

Figure 13 shows that an increased scale leads to a wider window in time but, at the same time, to a decreased frequency range. With the help of this mechanism, datasets containing high and low-frequency ranges can be analyzed. However, for analyzing only low frequencies, this algorithm may not be optimal because, according to G. Stockwell [41], the CWT may oversample the representation of the signal at low frequencies. Another disadvantage of the CWT is that it only has the power spectrum and amplitude, not the phase [45].

2.3.4.4 Hilbert-Huang Transform

The HHT was introduced by Huang et al. in 1998 for nonlinear and non-stationary signals [47–49]. Previous approaches like wavelets and STFT cannot address nonlinear signal analysis. The HHT is based on two parts [50]: The first and most important part is the EMD, which extracts intrinsic mode functions (IMF). The second step uses the extracted IMFs for a Hilbert spectral analysis.

The EMD extracts smooth envelopes from a time series *x(t)*. These envelopes are the IMFs. This decomposition assumes that the data may have several simple coexisting modes of significantly different frequencies at any time. These modes are all superimposed onto each other. To extract these modes (IMFs), a recursive algorithm is used. All local minima and all local maxima of the data series have to be identified in each iteration. Next, a cubic spline, which goes through all maxima, must be defined. This spline is the upper envelope. The same must be done with the minima to determine the lower envelope. The whole signal should be covered between the two envelopes. Then, the mean (see Figure 14) of these envelopes is subtracted from the initial data, and a new IMF is created.

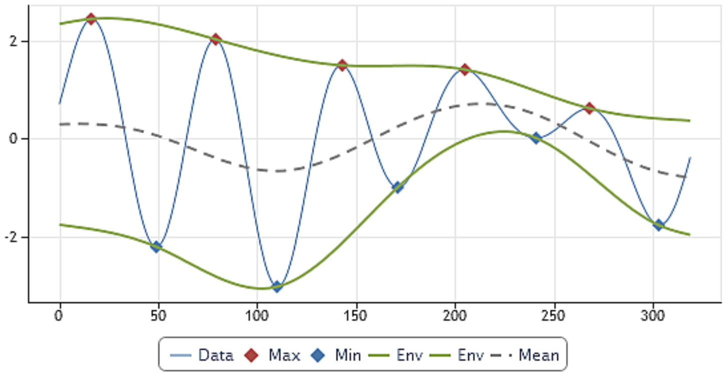

Figure 14: The minima (red) and the maxima (blue) envelope of the signal and their mean [50].

This extraction must be repeated *n* times until the remaining curve becomes a monotonic function, only has one extremum, or a stop criterion is reached. The remaining signal is called the residue.

After extracting all IMFs, the Hilbert spectral analysis is applied to compute instantaneous frequencies (see Section 2.3.3.2), where the amplitude and phase are a function of time.

The HHT process is not mathematically proven. It is only proven empirically by its practical application, which shows that it is a fitting solution for many scenarios [49]. Another downside of the HHT is the possibility of undesirable IMFs at low frequencies [51].

2.3.4.5 S-Transform

The S-transform was introduced by Stockwell et al. [52] in 1996. This was about the same time that the HHT was presented. Both intend to create a transform that can be used for calculating a time-frequency representation for non-stationary and nonlinear signals. The S-transform is based on the CWT. However, it enhances the CWT by being a frequency-dependent solution, providing a direct relationship with the Fourier spectrum—unlike a CWT whose result is in scales instead of frequencies. The S-transform of a function *h(t)* is comparable to a continuous wavelet transform with a particular mother wavelet that is multiplied by the phase factor [52]:

$$S(\tau, f) = e^{i2\pi f\tau} W(\tau, d) \tag{5}$$

Here, *d* is the inverse of the frequency *f*, τ is the time, and $W(\tau, d)$ is the CWT. The mother wavelet function *w* is defined by:

$$w(t, f) = \frac{|f|}{k\sqrt{2\pi}} e^{-\frac{t^2 f^2}{2}} e^{-i2\pi f\tau} \tag{6}$$

where *k* is a scaling factor that controls the time-frequency resolution. Since Eq.(6) does not satisfy the zero mean criteria required for a CWT, an S-transform is not a real CWT [52]. The complete formula of an S-transform is:

$$S(\tau, f) = \int_{-\infty}^{+\infty} h(t) \frac{|f|}{k\sqrt{2\pi}} e^{-\frac{t^2 f^2}{2}} e^{-j2\pi f t} dt \tag{7}$$

A significant advantage of S-transform is that it delivers higher precision in the frequency domain for low frequencies and, at the same time, higher precision in the time domain for higher frequencies, making it superior to the STFT [53].

2.3.5 Combined Analysis

The previously described feature extraction methods do not have to be used exclusively. Some approaches combine time-frequency analysis and frequency analysis with time-domain analysis methods. For example, a common feature extraction method is to use a feature extraction chain, which uses only the EMD part of the HHT as a first step. Afterward, the EMDs are analyzed with the help of time-domain analysis methods, as done by Kang et al. [34].

2.3.6 Comparison of the Different Methods

All three types of feature extraction (time domain, frequency domain, and time-frequency domain) are used in different predictive maintenance solutions [54]. As introduced in the previous sections, every category has its strengths and weaknesses. The important feature extraction methods and their previously described characteristics are listed in Table 2 and compared in the following.

Time-domain analysis methods are easy to implement and are, therefore, compatible with low-cost hardware. They are also used in combination with time-frequency analysis methods.

Most of the frequency domain analysis methods use FFTs as a first step. The benefit of these analysis methods is that they are an easy way to analyze signals for the appearance of particular frequencies. A drawback of frequency domain-based methods is that they always deliver a time-averaged spectrum. This may be good enough for stationary signals, but this method is insufficient if the signal changes over time. A possible solution is to slice the time series into smaller time windows and analyze them independently, as is done by an SFTF, which results in a time-frequency transformation. Whenever one needs to analyze variable signals, a time-frequency domain method should be used because it can deliver the characteristics for every point in time. The main features of each of the four most-used methods (STFT, CWT, HHT, and S-transform) are summarized below.

STFTs are easy to implement and provide good results for non-stationary linear signals of frequency ranges that are close to each other. However, they have the disadvantage that the information depends on the window size. This window size is fixed during the execution of the transformation. In addition, they are vulnerable to the leakage effect.

CWTs overcome both limitations of STFTs by using scaled and translated wavelets. However, this comes at the cost of a more computationally intensive algorithm. In addition, the phase data and the relationship to the frequency spectrum are also unavailable when using a CWT. Another disadvantage of the CWT is that it oversamples the signal representation at low frequencies.

The S-transform is an algorithm that provides phase and frequency data for nonlinear and non-stationary signals. It delivers higher precision in the frequency domain for low-frequency areas and, at the same time, higher precision in the time domain for higher-frequency areas.

The HHT is also usable for nonlinear and non-stationary signals. It is less resource-hogging than a CWT or an S-transform. However, there are also shortcomings (e.g., the possibility of undesirable IMFs at low frequencies). Finally, a significant disadvantage of the HHT is that it does not have a mathematical foundation. By now, the HHT has only been empirically proven.

Table 2: Summary of different feature extraction types.

Method	**Pros**	**Cons**
Time domain analysis	- Easy to implement, even on low-cost hardware	- Complex coherences are not detectable
Frequency domain analysis	- Good for stationary signals	- Only time average spectrum
Short-time Fourier transformation	- Extension to Fourier transform - Non-stationary data (limited)	- Time information depending on the window size - Invariable resolution
Continuous wavelet transformation	- Non-stationary data with a wide frequency spectrum	- No direct relationship to the frequency spectrum - No phase data - Resource-intensive algorithm
Hilbert-Huang transform	- Nonlinear and non-stationary signals	- EMDs are not mathematically proven - Possibility of undesirable IMFs
S-transform	- Nonlinear and non-stationary signals - Phase and frequency data	- Resource-intensive algorithm

2.4 Transfer Learning for Predictive Maintenance

2.4.1 Introduction

As described in the previous sections, collecting as much training data as possible is important for the usage of machine leaning. However, many machine learning scenarios still share in their lack of sufficient data for learning a model in their specific scenario. The technique of knowledge transfer tries to overcome this problem by using source domain data or a source domain model to enhance the model for a target machine learning task [55]. In general, there are two different approaches available. Transfer learning, on the one hand, covers the topics of domain adaptation and multi-task learning, and, on the other hand, incremental, reinforcement, and online learning [9]. Incremental learning aims to permanently and incrementally adapt to a new environment with knowledge from an existing model and a small number of samples from the new domain [16]. Instead, transfer learning aims to maximize the reusability of knowledge from an existing model to a new environment with limited information.

This thesis focuses on transfer learning. Therefore, in the following section, the definitions for transfer learning used in this thesis are introduced. Next, the thematic clusters of transfer learning are explained. Finally, transfer learning methods for CNNs and RNNs are shown, as well as the transfer learning loss functions they use.

2.4.2 Transfer Learning Definitions

In order to understand and apply the concept of transfer learning, it must first be defined. For this purpose, the definitions and notations of Pan and Yang [55] are used in this thesis.

Definition 1 – Domain and Task

A specific domain D can be described with the help of two elements: the feature space χ and the marginal probability distribution $P(\mathrm{X})$, where $\mathrm{X} = \{x_1, \dots, x_n\} \in \chi$. For a specific domain (expressed by $D = \{\chi, P(\mathrm{X})\}$), a task T is defined by the label space $Y$and the predictive function $f(\cdot)$. The term $T = \{Y, f(\cdot)\}$ is not observed but is learned with the help of training data pairs $\{x_i, y_i\}$, where $x_i \in X$ and $y_i \in Y$. This can also be written as probabilistic term $P(y|x)$.

If Definition 1 is mapped to a predictive maintenance example, then Y is a set of all labels (e.g., “good condition,” “defect”) and $f(\cdot)$ is a function that can predict the status (label) y_i of a collected sample of a component x_i, where x_i. is the measured value at the position *i* of all *n* recorded measured values X, which all are in the measuring range χ. Task T is the condition to monitor (e.g., the health of a component).

In the following, every artifact related to the source domain is denoted with subscript S, e.g., $D_S = \{ (x_{S_1}, y_{S_1}), \dots, (x_{S_{n_S}}, y_{S_{n_S}})\}$. For the target domain, where the subscript T is used accordingly, it is $D_T = \{ (x_{T_1}, y_{T_1}), \dots, (x_{T_{n_T}}, y_{T_{n_T}})\}$. Table 3 shows a summary of all these notations.

Table 3: Summary of the notations used for transfer learning.

Notation	Description
$\boldsymbol{\chi}$	Feature Space
$\boldsymbol{Y}$	Label Space
$\boldsymbol{T}$	Learning Task
$\boldsymbol{D}$	Domain data
$\boldsymbol{P}(\mathbf{X})$	Marginal distribution
$\boldsymbol{P}(\boldsymbol{Y})$	Label distribution
$\boldsymbol{P}(\boldsymbol{x}\|\boldsymbol{y})$	Conditional distribution
$\boldsymbol{f}(\cdot)$	Predictive function
Subscript S	Source domain
Subscript T	Target domain

With the help of the terms domain, task, source, and target, transfer learning can be defined according to Definition 2, which posits that transfer learning focuses on improving $f_T(\cdot)$ with the help of D_S [9]. In contrast to this, incremental learning focuses on improving $f_T(\cdot)$ with the help of T_S [56].

Definition 2 – Transfer Learning:

Transfer Learning is defined by a source domain D_S with its learning task T_S and a target domain D_T with the corresponding learning task T_T. Therein, the aim is to improve the predictive function in the target domain $f_T(\cdot)$ by using the knowledge of T_S and D_S. At this $T_S \neq T_T$ or $D_S \neq D_T$.

Research in previous years has mainly focused on "classical" transfer learning. However, deep learning is the dominating technique in many research fields today [57]. Here, a particular type of transfer learning called deep transfer learning is used. Tan et al. [57] used the following definition for deep transfer learning:

Definition 3 – Deep Transfer Learning:

A transfer learning task defined by $\langle D_S, T_S, D_T, T_T, f_T(\cdot)\rangle$ is called deep transfer learning when $f_T(\cdot)$ is a nonlinear function represented by a deep neural network.

A common problem regarding transfer learning, regardless of whether it is deep transfer learning or not, is that the influence of the source domain data can also have a negative effect on the training results in the target domain. This behavior is called negative transfer [58].

Definition 4 – Negative Transfer:

For a source domain D_S with its task T_S and a target domain D_T with its task T_T exist two different predictive functions $f_T(\cdot)$: a predictive function, $f_{T1}(\cdot)$, which is learnt only with the target domain data D_T and a second predictive function, $f_{T2}(\cdot)$, which is trained with the data of D_S and D_T. If the results of $f_{T1}(\cdot)$ are better than the results of $f_{T2}(\cdot)$, it is referred to as negative transfer, otherwise, it is called positive transfer.

Amongst others, a negative transfer can result from a conditional distribution difference between source and target [59]. In addition, the amount of labeled data in the target domain also influences the transfer learning process [60]. Therefore, one must weigh up the desired positive effect of a small amount of labeled data on the one hand, which can help to improve the feasibility and the reliability of finding a shared regularity between source and target. On the other hand, the result can worsen if there are only a few labeled target datasets because of overfitting. Using transfer learning with a large amount of labeled source data can also lead to negative transfer because the different source domain can impede the generalization [60].

2.4.3 Transfer Learning Thematic Clusters

2.4.3.1 Introduction

Transfer learning itself can be expressed by a definition, as has been done in the previous section. In addition, it can also be subdivided into different thematic clusters. These clusters are shown in Figure 15 and are described in detail in the following sections.

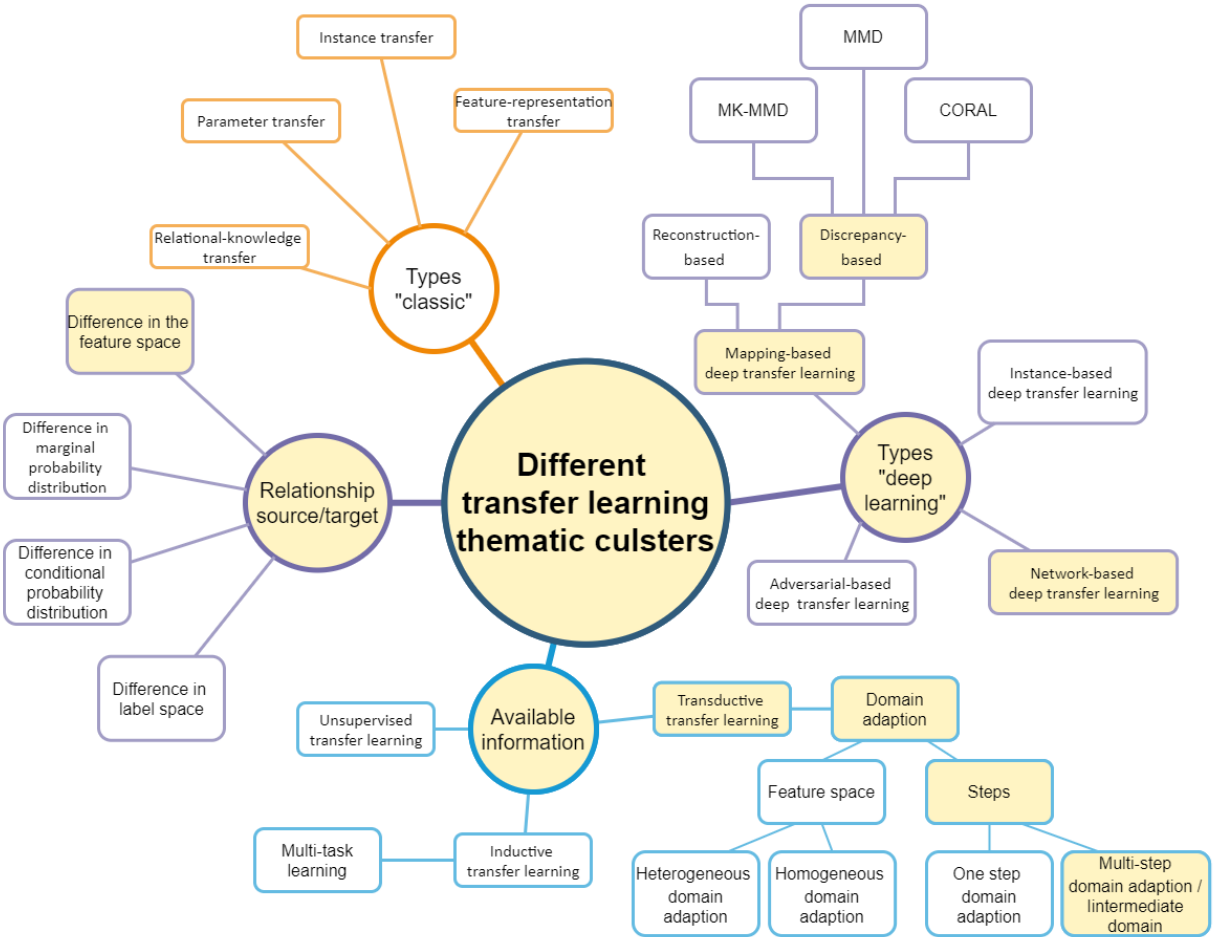
Instance transfer
Parameter transfer
Feature-representation transfer
Relational-knowledge transfer
Types "classic"
MMD
MK-MMD
CORAL
Reconstruction-based
Discrepancy-based
Mapping-based deep transfer learning
Instance-based deep transfer learning
Types "deep learning"
Adversarial-based deep transfer learning
Network-based deep transfer learning
Difference in the feature space
Difference in marginal probability distribution
Relationship source/target
Difference in conditional probability distribution
Difference in label space
Different transfer learning thematic culsters
Unsupervised transfer learning
Available information
Transductive transfer learning
Domain adaption
Feature space
Steps
Multi-task learning
Inductive transfer learning
Heterogeneous domain adaption
Homogeneous domain adaption
One step domain adaption
Multi-step domain adaption / lintermediate domain

Figure 15: Differentiation possibilities for transfer learning into thematic clusters. All transfer learning techniques can be differentiated by the available information, the relationship between a source and a target domain, and its transfer type. The techniques used in this thesis are highlighted.

2.4.3.2 Relationship of Source and Target Domain Data

Depending on the different relationships between the source and target domain data, it is possible to distinguish between four different use cases of transfer learning, all of them relevant for predictive maintenance [9]:

1. Difference in the feature space: $X_S \neq X_T$

   This is the case when the variables for the machine learning task change and can be:

   a. New sensors

   b. Different production processes

   c. Different machines

   d. Different data representation

2. Difference in label space: $\mathrm{Y}_S \neq \mathrm{Y}_T$

   In this case, nothing changes physically. Only the labels are changed. In the case of fault classification, this can be the change from a binary classification {“healthy,” “defect”} to a classification of the exact faulty component {“healthy,” “defect of component 1,” “defect of component 2”}.

3. Difference in marginal probability distribution: $P(\mathrm{X}_S) \neq P(\mathrm{X}_T)$

   This kind of transfer learning is needed for the following cases:

   a. Wear-out of sensors or machine components

   b. Changes in production process settings

4. Difference in conditional probability distribution: $P(y_S|x_S) \neq P(y_T|x_T)$

   This happens when the relationship between features and labels changes. A use-case for this scenario is when a machine produces the same product without any process changes but the product rejection rate changes.

2.4.3.3 Available Information

It is furthermore possible to differentiate between three types of transfer learning based on the information available in the source and target domain [55]:

- Inductive Transfer Learning: For this type, labeled data is available in the target domain. It can be distinguished between self-taught learning, which is the case when no labeled data is available in the source domain, and multi-task learning, when labeled data is available in the source domain and both tasks are learned simultaneously. For inductive transfer learning, the source and target tasks differ, while the domain can be the same or different.
- Transductive Transfer Learning: Here, labeled data is only available in the source domain. The tasks are the same in both cases. A distinction can be made between two cases: If the source and the target domain are different, it is called domain adaptation (DA). However, if there is only one single domain, it is called selection bias or covariance shift.
- Unsupervised Transfer Learning: In this setting, source and target tasks, as well as the domain, are not similar, and there is no labeled data available in either domain.

The transductive transfer learning approach of a DA can be further subdivided. Wang and Deng [61] categorized different settings of DA. The standard approach is a one-step DA, which can only be done under the assumption that the source and target domain are directly related. In this case, the transfer of knowledge can be done in one step. Wang and Deng further subclass the domain adaptation into homogeneous and heterogeneous DA. The feature space between the source and target domain in the homogeneous DA is identical. In heterogeneous DA, however, the feature spaces are different. If there is only a marginal overlap between the two domains or, in the case of a heterogeneous DA, the one-step DA may not be the best possible solution. Therefore, using an intermediate domain can bring the source and the target domain closer together to improve performance. This approach is also known as multi-step or transitive DA.

2.4.3.4 Classic Transfer Learning Types

It is also possible to differentiate transfer learning by the different transfer process types. There are four different types of performing a transfer learning task [55]:

- Instance transfer: The labeled data of the source domain or at least parts of it can be used for training in the target domain. Therefore, the source data must be re-weighted.
- Feature-representation transfer: This transfer type reduces the difference between the source and the target domain as well as the error of the resulting classification and regression models by finding a suitable feature representation.
- Parameter transfer: Here, the source and the target task share some of the domain-across hyper-parameters of their model. This contrasts with multi-task learning, where both tasks are learned in parallel.
- Relational-knowledge transfer: This transfer process type is used for transfer learning in relational domains where the data of the domains are not independent and identically distributed. In addition, the data can be represented by multiple representations, such as data in different social networks.

Not every transfer learning approach can be used for every transfer learning type. For example, relational knowledge transfer and parameter transfer are only used in an inductive transfer setting. However, the types can also be combined, as in the case of [62], which combines instance and parameter transfer.

2.4.3.5 Deep Transfer Learning Types

Previously, the “classic” transfer learning types have been explained (see Section 2.4.3.4). While the deep transfer learning approaches sometimes have different names in the literature, the idea remains the same. Tan et al. [57] described the following approaches for deep transfer learning:

- Instance-based deep transfer learning: This approach is similar to the classical transfer learning. Prominent approaches in the supervised context are AdaBoost-based techniques like TrAdaBoost [63], where the algorithm filters the source domain data to use only instances that are similar to the target domain. These instances are reweighted to have a similar distribution as in the target domain and are then used together with the data of the target domain for training the network.

- Mapping-based deep transfer learning: This approach has the same definition as feature-representation transfer in classical transfer learning. The target domain and the source domain are mapped into a new data space through which it is possible to train a network with both domains at the same time. An adaptation layer can do this mapping. In general, those algorithms measure the distance of probability distributions. The leading algorithm in this area is the MMD [64], which can be used directly as is done in a semi-supervised approach for translations by Hamilton [65] or in modifications like multi-kernel MMD (MK-MMD), which is used, for instance, by Tang et al. [66] for character recognition.
- Network-based deep transfer learning: Parts of a pre-trained network are reused in the target domain. This is comparable with the definition of parameter transfer in classical transfer learning. In the case of deep learning, the whole pre-trained source network, or at least parts of it, can be reused in the target domain neural network. In the case of classical transfer learning, only simple parameters are transferred. In deep neural networks, the front layers are typically used for the feature extractors of the more general aspects and are, therefore, well-suited for such a transfer. It is also possible to share the hidden layers, as Microsoft does in a language transfer setup [67].
- Adversarial-based deep transfer learning: Here, the adversarial technique is used to find transferable features for the source and the target domain. One way to do this is to add an adversarial layer to the network that gathers the data from both domains. This layer can be used to generate a domain adversarial loss, which is used in addition to the usual classification loss function [68].

Those approaches can also be mixed. This is especially the case for semi-supervised solutions like that of Tang et al. [66]. In their setup, as a first step, parts of a trained CNN of a source domain are reused for training with labeled target domain data (network-based deep transfer learning). Afterward, they use an MMD-based approach (mapping-based deep transfer learning) for unsupervised learning of unlabeled target data.

2.4.4 Transfer Learning for CNNs

The above-given theoretical transfer learning basics can be used for CNNs. Ample research is currently being devoted to this topic due to the fact that CNNs are a specialized type of artificial neural network with a primary focus on the classification of images, which is, for instance, important for internet-based technologies as well as for self-driving cars. Moreover, many commercial products from big companies like Tesla [69] or Microsoft [70] use CNNs in combination with transfer learning for classification tasks of images.

As described in Section 2.2.4, a CNN consists of three parts. An input layer, feature extraction layers (made of convolutional and pooling layers), and classification layers that are realized by fully connected layers. Each of them is important for transfer learning. There are three different transfer learning scenarios [27]:

1. Fixed feature extraction: The first two parts of a pre-trained CNN are reused. Only the fully connected layers must be retrained with the new data of the target domain. The weights and structure are frozen for the input and feature extraction layers. Therefore, this scenario is called fixed feature extraction.
2. Fine-tuning and frozen layers: Here, either the whole feature extraction part or at least the last layers are not frozen. The weights of these layers are fine-tuned with the target dataset. Since the earlier layers can handle more generic features, the number of layers for fine-tuning depends on the distribution of the target and the source dataset.
3. Pre-trained models: To avoid long training times, especially for large training sets, pre-trained models can be used. There exist several big models, such as Google LeNet [71], which can be used as initialization and fine-tuned afterward.

Determining which scenario should be used mainly depends on the target dataset size and on how different the source domain is from the target domain (see Table 4).

Table 4: Recommendation on how many layers should be fine-tuned depending on the dataset size and their similarities.

	Small target domain dataset	Large target domain dataset
Similar datasets	No fine-tuning of the feature extraction layers. Only modifying the last few fully connected layers to avoid overfitting.	The whole pre-trained model can be retrained with the new dataset. Because of the data size, the chances of overfitting are small.
Different datasets	The last few layers of the feature extraction layer should be fine-tuned to respect the different high-level features. The first few layers, which are for low-level features, should be frozen to avoid overfitting.	One way could be to train a completely new network. Nevertheless, it could be advantageous to initialize the new network with the pre-trained model.

The aforementioned scenarios apply to both supervised and unsupervised training approaches. For the supervised case, the process is straightforward: Use the new target domain data and retrain the network with the help of the available labels.

However, the domain adaptation process is more complicated for unsupervised learning, where different approaches exist. The most widespread solutions are mapping-based. These can be either discrepancy-based (mostly) or reconstruction-based. There also exist adversarial solutions.

Discrepancy-based approaches are similar in that the output of the source and target domain of one or more fully connected layers is used to calculate a domain adaptation loss. This loss is used to minimize the distribution discrepancy between both domains in the shared feature space (see Figure

16). For instance, this can be achieved by simply adding one MMD layer with a linear kernel, as is done by Tzeng et al. [72]. Furthermore, the intensity of the loss functions can be varied by multiplication of its value with a tradeoff parameter λ.

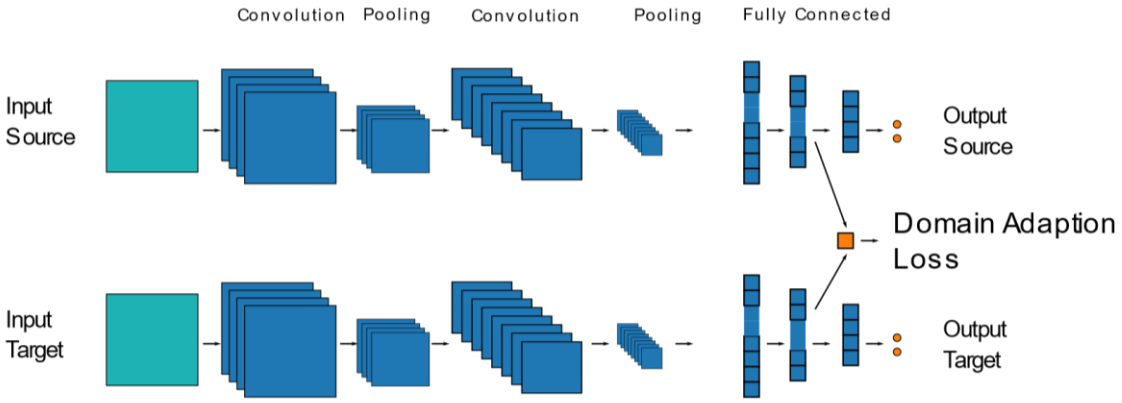

Figure 16: Transfer learning for CNNs based on domain adaptation loss. The output difference of the fully connected layers is used as input for the domain adaption loss function, which calculates the domain adaption loss.

In contrast to discrepancy-based approaches, reconstruction-based methods are not widely used. One example is that of Ghifary et al. [73], who train their network with labeled source data together with unlabeled target data. The model itself is based on an ordinary CNN for source label prediction and a deconvolutional network to reconstruct the target data.

The last used domain adaptation approaches are adversarial-based solutions like the one of Ganin et al. [74], who trained a network with two loss functions. The first is the ordinary loss for the class labels, and the second is a domain label loss. The aim is to maximize the label classification accuracy (for source samples), whereas the domain classification accuracy (for all samples) should be minimized through a gradient reversal layer. This results in the feature distributions of the two domains being made similar, leading to domain-invariant features. Tzeng et al. [75] proposed an similar approach that combined a generative adversarial network (GAN) loss with a discriminative model and unshared weights.

2.4.5 Transfer Learning for RNNs

A promising approach of using transfer learning for time series is to use RNNs and their LSTM and GRU derivates. However, in general, it can be said that there has been less research done in this area than around CNNs [76].

If both domains have labeled data, there are two ways of transfer learning. One way is to use fine-tuning by pre-training the network with the source datasets and afterward freezing all layers except the last few classification layers. This approach is similar to that described already in Section 2.4.4 for CNNs. The fine-tuning of RNNs for time series analysis is used by different researchers like Gupta et al.

[77], who used this approach for clinical time series analysis, or Yoon et al. [78], who used it for language processing.

Another approach for labeled data is multi-domain learning by using both datasets together. Toby Perrett and Dima Damen [79] presented a Dual-Domain LSTM, which optimizes this approach by using batch normalization on the input-to-hidden and the hidden-to-hidden weights of the LSTM. They evaluated their approach to datasets for cooking-related activities.

If there is no labeled data, there are mainly two research directions in the context of domain adaptation for RNNs. The first is to reduce the difference between the two domains with techniques such as MMD, as done with CNNs. This can be done using a deep recurrent neural network with fully connected layers at the end. This approach has been used for a natural language processing setup with a Sina Weibo dataset (a large Chinese microblogging website) by Xiao Ding et al. [80]. They used a Tree-LSTM, which is a modification of the LSTM that processes inputs—not in a chain-based manner, as in the original implementation, but in a tree-based manner to reflect sentences in natural language. Then, the weights of the supervised trained model are transferred to the domain adaption model. Finally, the domain adaption is done using a modified MMD called a tree kernel-based MMD (TK-MMD).

Another popular concept is using deep adversarial-based RNNs. This is based on feature extraction, domain classifiers, and label predictors. For feature extraction, an LSTM can be used. A particular focus is given to label prediction, which synchronously uses both datasets (labeled source and unlabeled target) to learn the domain labels. For instance, this approach is used by Da Costa et al. [81], who used classical LSTMs, as well as by Liu and Gryllias [82], who used Bi-LSTMs for feature extraction. A Bi-LSTM is an LSTM, which not only takes the output of the previous time point, but also the output of the next time point as input.

2.4.6 Transfer Learning Loss Functions

2.4.6.1 Introduction

In the previous chapters, transfer learning for CNNs and RNNs has been described in detail. The most frequently used transfer types for unsupervised and semi-supervised domain adaption are mapping-based approaches. As such, discrepancy-based algorithms are used almost exclusively (see Section 2.4.4). All these algorithms are similar in that they compare the output of fully connected layers of the source and the target domain. The more significant the difference between the output of the two domains, the larger the result of these algorithms is. In addition, they result in zero if the distributions are identical, which matches the requirement of most training algorithms that try to minimize the loss [6].

The most common loss functions are MMD, Wasserstein, and CORrelation ALignment (CORAL) [54, 81, 83]. A particular position among these algorithms has MMD, which is not only used directly but in a variety of variations, such as MK-MMD [84], joint MMD (JMMD) [85], and TK-MMD [80]. Amongst all MMD variations, MK-MMD is the most frequently used. MMD, MK-MMD, Wasserstein, and CORAL are discussed in the following sections. They are also used in Section 7.2 as a reference to measure the here-developed transfer learning approaches.

2.4.6.2 Maximum Mean Discrepancy

The MMD uses the mean discrepancy to measure the distance between two distributions and is defined by Gretton et al. [64] according to Eq. (8).

$$MMD_b(p,q) \ = \sqrt{\frac{1}{n_s^2}\sum_{i,j=1}^{n_s} k(x_i^S,x_j^S) - \frac{2}{n_s n_t}\sum_{i,j=1}^{n_s,n_t} k(x_i^S,x_j^T) + \frac{1}{n_t^2}\sum_{i,j=1}^{n_t} k(x_i^T,x_j^T)} \tag{8}$$

p and q are the distributions of the source and target domain. The source domain samples are given by $x^S$and its number, n_s. For the target domain, x^T and n_t are used accordingly. The kernel function k is used for mapping the values in a Reproducing Kernel Hilbert Space (RKHS) (see also He and Ding [86]). Normally, a Gaussian kernel function (Eq. (9)) is used. The parameter γ defines the width of the Gaussian.

$$k(x^S,x^T) \ = e^{-\gamma\|x^S-x^T\|^2} \tag{9}$$

The Gaussian kernel is a differentiable function, which also makes MMD differentiable. This allows it to be used as a loss function for training types based on gradient descent [65]. When using MMD as a loss function for transfer learning of CNNs, x_i and x_j are the outputs of the fully connected layers (see Section 2.4.4).

MMD has been used successfully in many transfer learning setups [87]. However, a minor drawback emerges if the distribution of the classes is very different. In this case, MMD may not be an optimal choice. An example of this is using the well-known handwriting dataset MNIST as a source dataset and a dataset of house numbers as a target dataset. In the target dataset, the number 1 would occur disproportionately often since there are hardly any streets without the number 1. However, the number 0 occurs relatively rarely [87].

2.4.6.3 Multi Kernel MMD

The choice of the kernel has a significant impact on the performance of MMD. The reason for this is that a different kernel may embed probability distributions into a different RKHS that brings out the sufficient statistics differently. Therefore, the proper selection of the kernel leads to the optimal effectiveness of the system. By using MK-MMD, the impact of a wrong kernel can be reduced by employing a mixture of multiple kernels [12]. MK-MMD is defined as given in Eq.(10).

$$k\left(x_i^S, x_j^T\right) = \sum_{m=1}^{K} k_m(x_i^S, x_j^T) \tag{10}$$

where k_m (m = 1, ..., *K*) represents one of the *K* different kernels. For instance, when using a Gaussian kernel according to Eq. (9), each kernel has a different γ. The different source and target domain distributions are represented by x_i^S and x_j^T, respectively.

2.4.6.4 Wasserstein Distance

The Wasserstein distance can measure the distance between two probability distributions [88]. Its origin is a loss function for the training of GANs. However, the Wasserstein distance can also be used as a loss function for transfer learning in a domain adaptation case [83].

The definition of the Wasserstein distance is based on a compact metric set $\mathfrak{H}$. Prob($\mathfrak{H}$) denotes the space of probability measures on this compact metric set. The Wasserstein distance for two distributions *p, q* ∈ *Prob($\mathfrak{H}$)* is determined as given in Eq. (11).

$$W(p,q) = \underset{\mu \, \in \, \Pi(p,q)}{inf} \quad \mathbb{E}_{(h^s,h^t)\sim \, \mu}[\Vert \; h^s - \; h^t \; \Vert] \tag{11}$$

μ stands for a specific joint probability distribution. Π(*p, q*) represents the set $\mathfrak{H}$ x $\mathfrak{H}$ of the available joint distributions *μ(h^s, h^t),* whose marginals are *p* and *q*. This algorithm results in the calculated optimal transport plan *μ(h^s, h^t),* which indicates how much 'mass' must be transported from domain $\mathrm{h^s}$ to domain $\mathrm{h^t}$ to move the distribution *p* into *q*. Therefore, the probability distributions can be seen symbolically as two heaps of a certain mass of earth and the Wasserstein distance as the minimal cost for transferring one of the heaps to the other. Because of this, the Wasserstein distance is also known as the earth mover's metric [89].

2.4.6.5 Deep Correlation Alignment

Another loss function is the CORrelation ALignment (CORAL) [90], which aims to align the second-order statistics (namely, the covariance) of the source and target domain features. The domain discrepancy loss ($\mathrm{L_{CORAL}}$) is calculated as given in Eq.(12),

$$L_{CORAL} = \frac{1}{4d^2} \Vert \; C_S - \; C_T \; \Vert_F^2 \tag{12}$$

where $\Vert_F^2$ is the squared matrix Frobenius norm, and *d* denotes the dimension of the activations. The covariance matrix of the target and source domain features are $\mathrm{C_T}$ and $\mathrm{C_S}$ and are defined by:

$$C_S = \frac{1}{n_S - 1} \, (I_S^{\mathrm{T}} - \frac{1}{n_S} \, \left(\mathbf{1}^{\mathrm{T}} I_S\right)^{\mathrm{T}} \left(1^{\mathrm{T}} I_S\right)^{\mathrm{T}} \tag{13}$$

$$C_T = \frac{1}{n_T - 1} \, (I_T^T - \frac{1}{n_T} \, (\mathbf{1}^T I_T)^T (1^T I_T)^T \tag{14}$$

where I_S and I_T are both input data. This data is the previous layer's output in a deep learning network of the source (X_S) and the target (X_T) domain data. n_S and n_T should be equal and are the number of samples of both domains. **1** is a column vector with all elements set to 1.

2.5 Conclusion

There are many different techniques available for machine learning. This concerns the preprocessing by different feature extraction methods and the actual machine learning algorithm. Here, the best-fitting solutions have to be chosen. Some techniques can be used for supervised learning only, and others, such as CNNs, can also be used in unsupervised setups. Some can be used for regression, and others for classification tasks. Especially in the context of predictive maintenance, where the analysis is often based on sensor signals, it is advisable to prepare the data by a suitable feature extraction method independent of the ML technique used. For this purpose, algorithms in the time, frequency, and time-frequency domains are available. In the field of predictive maintenance, it is often the case that the training dataset only consists of a few samples. These samples are also often unlabeled. For this purpose, the domain adaptation method is useful to take additional knowledge from a second dataset into account.

Regardless of whether it is a classification or a regression problem, it must be decided which techniques are the most suitable for the given task. For instance, classification tasks can be handled by various approaches such as CNN or SVM. However, not all approaches can handle domain adaptation very well. In the context of domain adaptation with deep learning techniques, different approaches are available that are mainly based on different domain adaption loss functions. As such, the matching loss function has to be taken.

Regression tasks can also be handled by different approaches like SVRs and RNNs, which can handle the time dependencies of the input data. When using a deep learning approach, such as an RNN, for a domain adaptation, the question of the most appropriate loss function arises again, as it did for the classification approach. The predictive maintenance regression tasks mainly concern the prediction of the remaining useful life/time to failure scenarios.

This chapter has not only introduced the machine learning background, but it has also contributed content to RC1 and RC2. The points discussed can be assigned to the following research challenges. RC1, which asks for appropriate methods for predictive maintenance tasks of machines based on features of sensor data, is touched on by Section 2.2, Section 2.3, and Section 2.4. Section 2.2 presented the most important machine learning techniques, which can be used for condition monitoring of sensor data. These are CNNs, RNNs, and SVMs. This was followed by Section 2.3, which presented feature extraction techniques that can be applied to raw sensor data. Here, a particular focus was on the techniques of all three domains (time domain, frequency domain, and time-frequency domain). Section 2.4, which handles the basics of transfer learning, is not directly involved with the sensor data but with the effects of limited sensor data. Here, transfer learning can provide a solution. Section 2.3.6 compares the different feature extraction methods for sensor data and describes their constraints. This directly targets RC2, which asks for the constraints of the different techniques.

3 Predictive Maintenance

3.1 Introduction

Predictive maintenance is a maintenance process based on the evaluation of process and machine data and is found primarily in the context of Industry 4.0. By processing the underlying data, forecasts become possible that form the basis for needs-based maintenance and a consequent reduction of downtime [1]. In order to gain a precise understanding of this process, this chapter will first incorporate predictive maintenance into the field of Industry 4.0. This chapter further presents two use cases for transfer learning in the predictive maintenance environment. The first is a fault classification task, and the second is a useful life prediction task. Both are applicable for bearings in grinding spindles. Therefore, the technical backgrounds of machine tool spindles, bearings, and accelerometers are introduced. Next, the current state of the art for predictive maintenance for classification and the remaining useful life of bearings is presented. This is done in order to provide an overview of what is lacking within the current approaches and how they can be improved, which is also done to tackle RC3. This also results in an overview of what is missing in current approaches and how they can be improved. Based on this overview, the three research questions of this thesis are derived.

3.2 Predictive Maintenance Background

3.2.1 Introduction

The vision of a smart factory (fully connected and self-organizing) in the industrial environment is only possible if the machines used work correctly and deliver results with the desired efficiency and precision [1]. One way to achieve this is through the usage of predictive maintenance. Therefore, in this chapter, predictive maintenance is first brought into the context of Industry 4.0. Subsequently, spindles, an essential component in the machine manufacturing process, are introduced [3]. Within these spindles, the most critical element is the bearing [4], which is presented in detail in this section. Finally, accelerometers, which are used for predictive maintenance tasks, are introduced.

3.2.2 Industry 4.0

Predictive maintenance relies on a sequence of digitalization stages and is a subarea of Industry 4.0 (I4.0). I4.0 denotes the fourth industrial revolution and aims to automate industrial practices and traditional manufacturing with the help of smart technologies such as machine-to-machine communication and the internet of things [91, 92]. A group of researchers and industry experts has developed an I4.0 reference model for a better understanding of the process [93]. The I4.0 development path can be structured along six stages (see Figure 17): The first two stages are in the field of pure digitalization. Everything is based on stage 1, the “computerization” stage, where tasks are performed with the help of computers. This involves business tools like enterprise resource planning (ERP) systems as well as production machines with a computerized numerical control (CNC). The next step, "connectivity," connects the different isolated components. This is usually achieved by connecting the components in a local network. The remaining four stages are in the I4.0 field. The third stage is "visibility," which focuses on presenting what is happening. To do so, sensors must be placed on the machines to record the measured values throughout the entire process. These measurements can be used to create a digital model that shows the current state of the processes. The next phase is "transparency." Companies at this stage use the data gathered in phase 3 to understand why something is happening. This is followed by a step called "predictive capacity," wherein algorithms use the gathered data to predict what will happen in the future. The last stage is "adaptability," an autonomous process in which decisions are taken to optimize the process without human interactions. All these stages build upon each other directly. Therefore, it is impossible to move up to a higher stage if the current one is not fulfilled.

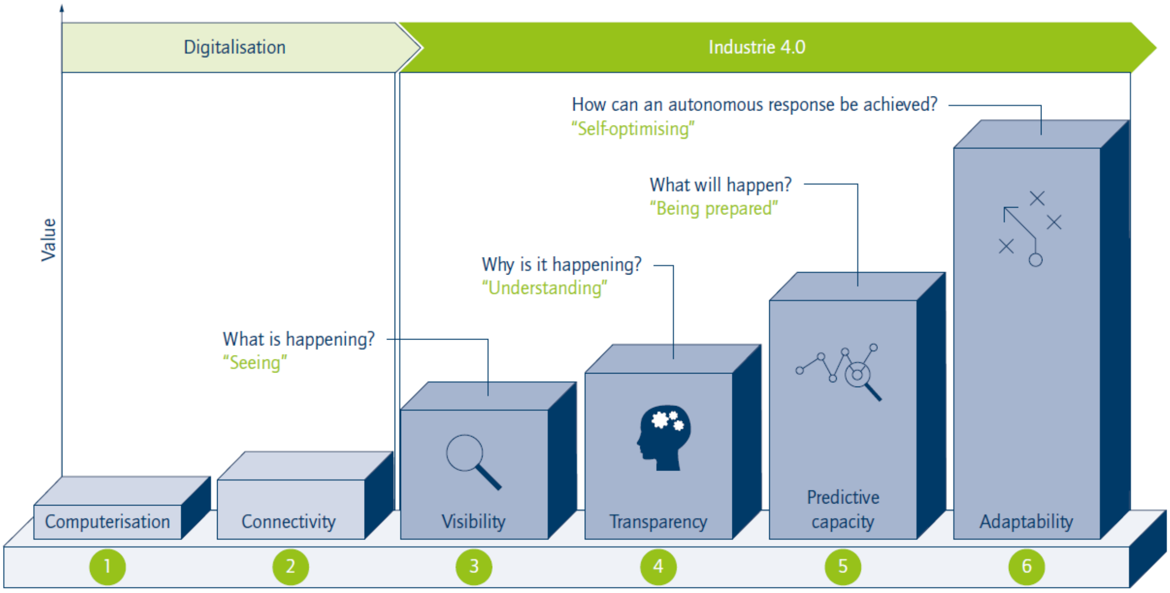

Figure 17: Stages in the Industry 4.0 development path. The path begins with stage 1 computerization and ends with stage 6, which leads to autonomous adaptions [93].

Applying this definition of I4.0 to predictive maintenance shows that companies must be at least at stage 5 to cover all predictive maintenance aspects. This is because predictive maintenance, which is understood as the proactive maintenance of machine tools, consists, amongst others, of error detection of known errors, which is stage 3, and error prediction, which is stage 5 [94].

In order to gain an essential business understanding of predictive maintenance in general and bearing damage in grinding spindles in particular, machine tool spindles and bearings are explained in more detail in the following two sections. Parts of these sections have already been published in a research article by Schwendemann et al. [54].

3.2.3 Machine Tool Spindles

Today, there are numerous different types of machine tools. However, they all share the need to remove material somehow. This is done using a cutting tool attached to a spindle, which in turn is responsible for the precision and speed of the machine. It absorbs the emerging cutting forces and guides the tool. A spindle is usually driven directly by an electric motor but can also be driven indirectly by a belt drive [95]. To be more precise, the tool is attached to the shaft of a spindle, which is clamped to the spindle in two positions: the front bearings at the front, which absorb the cutting forces, and the rear bearings at the backside that hold the shaft at position [96] (see Figure 18). The spindle is positioned on the workpiece by linear and rotational axes. One particular type of machine tool is a grinding machine, which removes the material with a grinding wheel consisting of hard material grains attached to a basic body using a binding material [97].

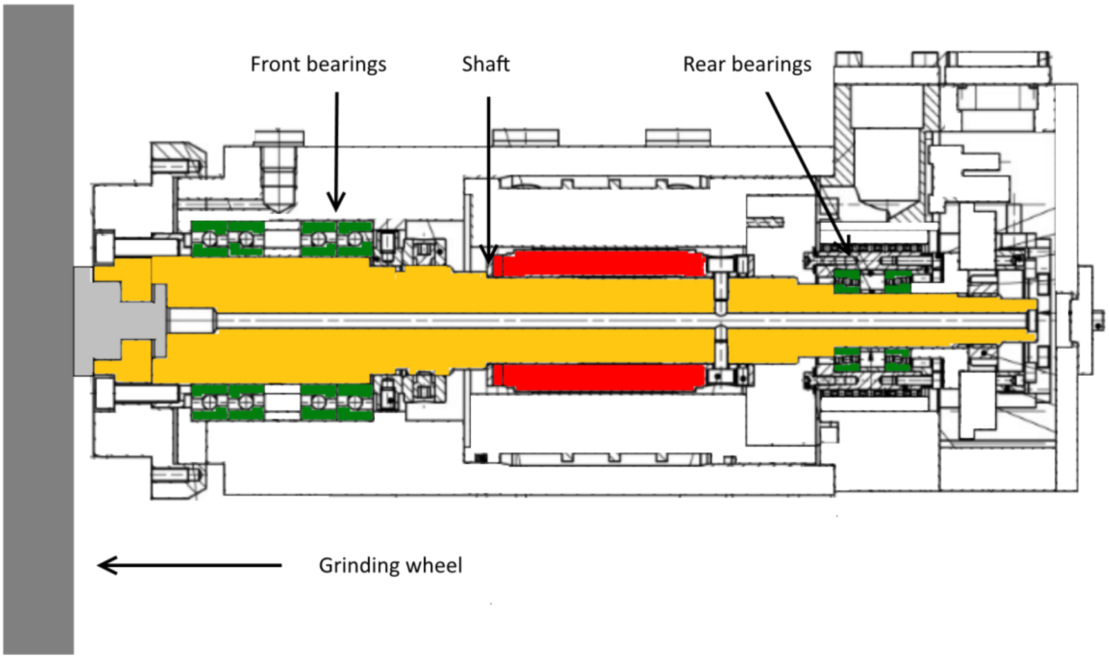

Figure 18: Positions of bearings (green) inside of a grinding spindle.

Even within the subcategory of grinding machines, there are a large variety of different spindles due to the specific requirements of the grinding tasks. There exist special grinding spindles for powertrains that need high power because they use big and heavy grinding wheels to remove much material. Because of the large wheel diameter, the circumference speed is high, even with a low rotational speed. However, there are also spindles for smaller components that need less power but higher rotational speed (see Figure 19) [98].

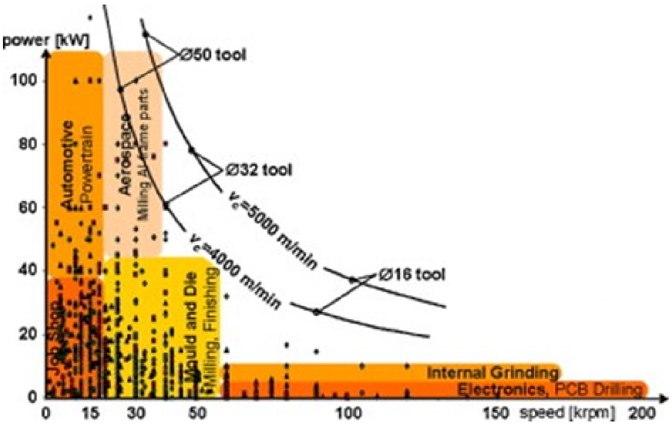

Figure 19: Different available spindle configurations and their applications [98].

3.2.4 Bearings

All the bearings inside spindles have the same layout, independent of their use case, like the grinding of a powertrain. This is based on their application purpose, which is to connect the rotating spindle shaft to the stationary spindle housing. Therefore, bearings are made of four components: the inner and the outer rings, the rolling elements, and the cage. For the use case of spindles, most bearings use balls as rolling elements [99]. The cage keeps the balls at a constant distance from each other. As shown in Figure 20, the cage and the balls are placed between the inner and the outer ring. Each of the four components can be the source of a bearing fault. However, it is supposed that 90% of all faults are related to outer and inner ring defects [100]. A possible reason for this could be that the rings are permanently under load while the balls rotate, and therefore their contact area is constantly changing. However, the cage does not have to hold any load at all—it just has to keep the balls at a distance [101].

Each of the four components (outer ring, inner ring, balls, and cage) has a specific fault frequency (Eq. (15) – (18)), which is emitted by the balls that roll over the surface of the inner and outer rings [101–104]. Depending on the rotational speed, there are use cases where these frequencies are lower than 85 Hz [105]. Each anomaly of a component results in a periodic impulse that depends on the rotational speed of the shaft f_r and the geometry parameters. The relevant parameters are the pitch diameter (D_m), the number of balls (N_{balls}) and their diameter (d), and the contact angle (∝) (see Figure 20). The contact angle defines the angle of the contact position between the rings and the ball. An angle of 0° stands for a vertical contact line.

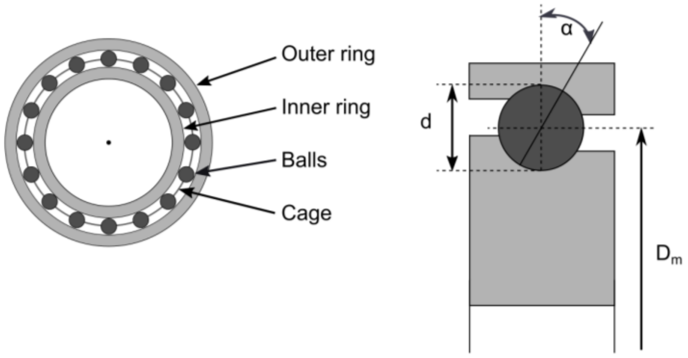

Figure 20: Layout of a ball bearing in the front and side section plane [54].

$$f_{cage\ fault} = \frac{1}{2} * f_r * (1 - \frac{d}{D_m} * cos \propto) \tag{15}$$

$$f_{outer\ ring\ fault} = \frac{1}{2} * f_r * \ N_{balls}(1 - \frac{d}{D_m} * cos \propto) \tag{16}$$

$$f_{inner\ ring\ fault} = \ \frac{1}{2} * f_r * \ N_{balls}(1 + \frac{d}{D_m} * cos \propto) \tag{17}$$

$$f_{ball\ fault} = \ \frac{1}{2} * f_r * \frac{D_m}{d} * (1 - (\frac{d}{D_m} * cos \propto)^2) \tag{18}$$

The aforementioned faults can occur in the following ways:

- Single-point defects: In this case, one of the four components of the bearing has a defect at a single point. All other bearing components are in good condition. Some examples of these defects are spalls, cracks, and pits. For this type of defect, an increased amplitude of the characteristic fault frequency of the given defect can be measured (see Eq. (15) – (18)). Most of the recent research articles address single-point faults [106, 107].
- Multiple-point defects: This term is used when a bearing has multiple single-point defects. Depending on the position of the defects, the amplitudes in the frequency range may vary. The amplitudes can sum up or compensate each other [108].
- Distributed faults: In contrast to point defects, this type of fault can result from the loss of lubrication, coupling misalignment, or contamination. This fault results in the bearing surface becoming rougher and rougher, which is why it is also called "generalized roughness." This leads to the fact that, unlike multiple-point faults, the fault cannot be subdivided into a series of multiple single-point faults. This, in turn, results in the problem that the characteristic frequencies are not necessarily measurable or might not even be present at all [107].

In the event that a bearing is degraded due to a short-term load, such as a machine crash, the characteristic frequencies appear immediately. However, the degradation of a bearing usually consists of four stages, as can be seen in Figure 21 [109, 110]. During the first stage, a crack is just developing and is only visible through ultrasonic frequencies. Therefore, neither characteristic fault frequencies nor visual changes of the bearing are visible. In the next stage, the crack continues to increase, although still not visible on the surface. However, the crack is now measurable through the natural frequencies of the bearing components. These natural frequencies become apparent because the forces that now arise are strong enough to excite them [111]. They are usually in a frequency range of between 2 kHz and 6 kHz [110]. In stage 3, the degradation becomes visible. The crack evolves into defects on the surface of the bearing. Small parts of the defective bearing component may come off. These surface changes lead to the appearance of the characteristic fault frequencies already described above. The last stage occurs directly before the total failure of the bearing. The clearance inside of the bearing increases. In addition, some severe pits may be fixed with removed parts from other pits and afterward smoothed over by the rolling elements. These two occurrences can lead to a vanishing of the characteristic fault frequencies. Instead, random frequencies appear in the form of background noise. This noise is in the frequency area of the characteristic fault frequencies and of the natural frequencies.

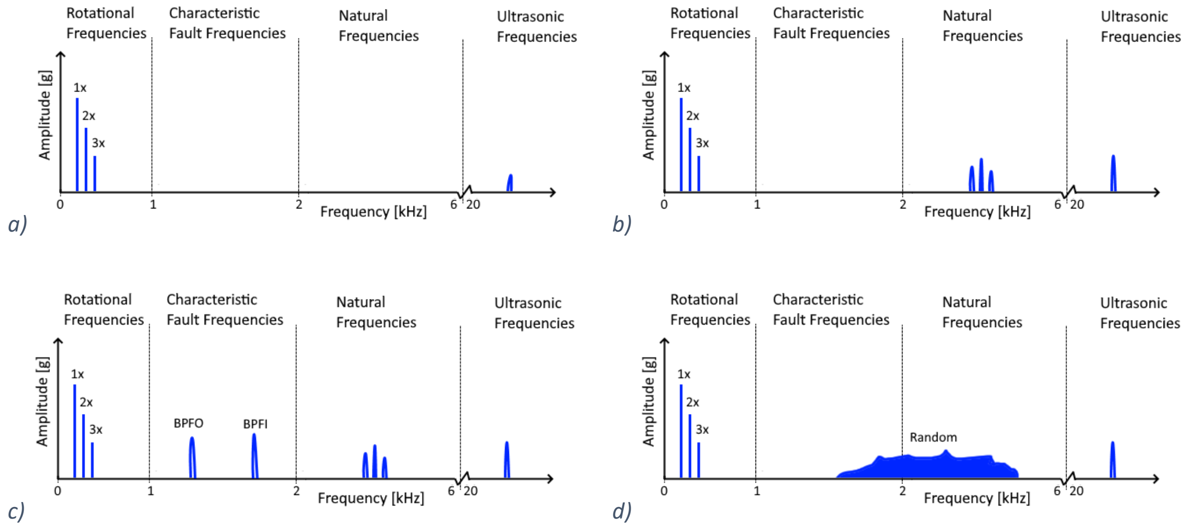

Figure 21: The four degradations of a bearing. In every stage, the rotation frequencies are visible in area A. The first stage is shown in a). Here the crack is only visible through ultrasonic frequencies. In stage 2 (b), the natural frequencies of the bearing become visible. This is followed by stage 3, where the characteristic fault frequencies appear in addition to the two already existing frequencies. Finally, in stage 4 (d), the frequencies in areas b and c vanish. Instead, random noise appears [112].

There are guideline values for the occurrence of the four stages in percentage terms of the L_{10} bearing lifetime (see Table 5) [109]. The L_{10} lifetime is the lifetime reached by 90% of the bearings of one type [101]. The current industrial solutions are condition monitoring and predictive maintenance systems with a primary focus on stage 3 [35]. This is probably because in order to prevent a machine breakdown, it is not necessary to detect a bearing fault earlier. Therefore, this thesis also focuses on faults in stages 3 and 4. An example of having enough time to perform a planned maintenance is the bearing in the workpiece spindles of grinding machines. Here, an operating life of 20,000 to 30,000 hours is assumed [101]. According to Eq. (19), this leads to a remaining lifetime of 12.5 weeks until the spindle bearing fails. This calculation is under the assumption that the machine is operated in two shifts and only on weekdays, in conjunction with the assumption that stage 3 is reached at 5% of the service life (as shown in Table 5).

$$Remaing\ Lifetime = \ 20000 \text{ hours} * \frac{days}{16\ hours} * \frac{weeks}{5\ days} * 5\% = 12.5 \text{ weeks} \tag{19}$$

Table 5: The occurrence of the different degradation stages in percentage terms of the L_{10} bearing lifetime [109].

Stage	1	2	3	4
Percentage of L_{10} lifetime [%]	10 – 20	10 – 5	5 – 1	1 – 0

Many publications focus on detecting the defects of bearings [102, 113, 114]. For some, only the fault state is relevant, and others also rely on the aforementioned degradation. In general, all the research works can be divided into two research directions. One direction is the determination of the physical size of the defect (e.g., the size of the hole in the inner ring). The other direction is the current degradation level of the bearing.

Because the characteristic fault frequencies only depend on bearing parameters and the rotational speed (see Eq. (15) - (18)), the research findings can generally applied to bearings in spindles. However, when analyzing bearing faults, it is not only the bearing itself that is relevant but also its environment. For the use case of a bearing inside a spindle, noise is the most important factor to consider. Other components of the machine, such as other spindles, cooling systems, or even frequency converters, may be the source of this noise. Typically, this noise is outside of the bearing's frequency band. There are three approaches to handle noise: 1: utilize only the frequency bands in which bearing defects can appear [103]; 2: de-noising of the data before using it [115]; 3: use noise-resistant algorithms [116].

3.2.5 Accelerometers

Accelerometers are used to record the characteristic fault frequencies mentioned in the previous chapter. There are two measuring principles of accelerometers used in the industrial environment: MEMS (micro-electro-mechanical system) and piezoelectric sensors. MEMS sensors are less expensive and have high shock resistance. The sensors based on piezo electronics, on the other hand, are known for their low noise and high linearity [117].

The accelerometers used in today's industrial environment can record data with a maximal resolution between 10 kHz and 50 kHz. However, most of them are only capable of recording up to 10 kHz [118, 119]. Even within many research paper datasets, only sensors up to about 25 kHz are used (e.g., 25.6 kHz [120] and 12 kHz [121]).

When recording accelerations of bearings, the following should be considered. To be able to record all possible fault positions, at least two sensors that are aligned to each other at right angles are required. This is because one part of the bearing (inner ring or outer ring) is fixed and operated under radial load. Not all possible positions of the fixed part can be recorded with a sensor that only records in one direction. Under misalignment of the bearing, all three dimensions are needed [37]. However, often, only one sensor is used because even MEMS sensors are expensive, and it is possible to cover large parts of the bearing with only one sensor.

One way to fit the industrial demands for having a good return on investment is to use triaxial sensors to avoid the cost of multiple sensors and their wiring. Triaxial sensors record all three axes with only one sensor. Their usage is feasible because of their assembly, which uses three small accelerometers combined in one housing. However, their applicability is limited since sensors intended for industrial use are only available with approximate sampling rates of 5 kHz (e.g., 4.5 kHz [122] and 5.5 kHz [123]). The usage of accelerometers to measure vibrations is the most common way to analyze the status of a bearing [124, 125]. Nevertheless, there are also investigations regarding the actual and historic temperature profile [126], the mechanical forces occurring, such as preload in the bearing [127], and the motor current and voltage [128]. However, vibrations are found to be the most reliable source for the condition of a bearing [129]. Additionally, a DIN standard exists that describes the proper application of accelerometers for bearing monitoring [124].

3.2.6 Challenges

This section has introduced the technical background for predictive maintenance in general. In addition, a particular focus was set on bearings inside of spindles, whose predictive maintenance tasks have been in focus for a long time due to their importance in the production process. The physical background described above can be used in combination with traditional methods like simple threshold value monitoring in order to detect errors [130]. However, those traditional approaches are challenging in that frequency bands, harmonics of the frequency, and the relationship between them must also be considered for a more accurate analysis. Therefore, experts with a deep knowledge of the degradation process are needed. Nevertheless, even for those experts, it is nearly impossible to identify all the convoluted features that can appear when multiple-point defects or distributed faults of a bearing appear [131].

As introduced in Section 2.2.1, machine learning can be used to overcome the problems described above. Like traditional approaches, they somehow create a mapping of vibration signals to classify them into different categories like “healthy” or “outer ring defect.” The difference is that when using traditionally approaches, the classification is carried out using hard empirically determined limits for particular features. When using machine learning, the classification is based on trained models, which rely on often labeled data. A trained model involves more features than a simple hard limit (e.g., [34]). These assumptions can be directly applied to the case of estimating the RUL as well, since during the RUL estimation also the current health state is used to predict the remaining lifetime [132].

The downside of ML approaches is that they need a lot of data for training their model (see also Section 2.2.1). Especially for the use case of bearings, this is not feasible for real-world approaches. Often, small and unlabeled or partly labeled datasets are available for these cases. This is the case for new machines or new types of a component [9]. However, existing machines also suffer from this problem for the following three reasons [131]: First of all, it can be dangerous when machines are running with faulty components; therefore, they are often replaced in advance. Second, the degradation process of bearings is really slow. Therefore, it takes a lot of time to get to a faulty state (see also Section 3.2.4). Third, the different process conditions of a machine might produce different error patterns. A solution to overcome the lack of datasets is to use transfer learning to use datasets of a different bearing or a different process condition [12, 83, 131]. Therefore, the next section introduces the state of the art for transfer learning for predictive maintenance of bearings.

3.3 State of the Art for Transfer Learning for Predictive Maintenance of Bearings

3.3.1 Introduction

The previous chapter has shown the importance of using machine learning technologies and transfer learning for the use case of predictive maintenance for bearings. As introduced in Section 1.1, there are two main research directions: fault classification and the estimation of the RUL. Therefore, the following sections present a review of selected research regarding classification as well as for RUL in the context of bearings. In each case, supervised learning approaches are discussed first. After that, transfer learning approaches for the same bearings are presented, followed by approaches for different bearings. A particular focus is placed on transfer learning approaches and their strengths and weaknesses. This results in the derivation of the research questions of this thesis.

Most parts of the following sections have already been published in research articles by the author [54, 112, 133].

3.3.2 Bearing Fault Classification

3.3.2.1 Introduction

Bearing fault classification is a field of great interest. Therefore, many approaches using different techniques exist. There are approaches that use traditional machine learning, where often a small number of features are extracted, which are then used for algorithms like SVMs [113, 134, 135], as well as approaches that use deep learning techniques, such as CNNs, where often no separate feature extraction is performed [136, 137]. In addition, different classification tasks like fault location (inner ring, outer ring, etc.), fault size, and fault severity exist.

Literature reviews reveal two important facts about current research on bearing fault classification [54, 131]: First, most publications are based on only a few reference datasets, and second, CNNs can deliver the highest accuracies for all classification tasks. Accordingly, the reference datasets, as well as the current state of the art based on CNNs, are presented below.

3.3.2.2 Reference Datasets

Two publicly available datasets are used in most publications. The one used in most of the current research articles is that of the Case Western Reserve University bearing data center (CWRU) [121]. It provides data from a two-horsepower electric motor. The recording of the signals of the bearings, which support the motor shaft, is done with accelerometers on both sides of the motor: drive-end (64 measurements of vibration signals) and fan-end (49 measurements of vibration signals). All faults are artificial single-point faults seeded in the balls, the outer ring, or the inner ring of the bearing. This is carried out with an electro-discharge machining operation. There is a variation of faults by different fault diameters and different motor loads from a horsepower of 0 to 3. Please refer to the descriptions of Loparo [121] for more information.

The University of Paderborn supplies another publicly available dataset [36]. It includes measurements of artificial defects in the outer race, the inner race, and the balls of the bearing. In addition, it includes data of real damages caused by accelerated lifetime setups. The entire dataset consists of 32 different bearings of the same model. Data is recorded by operating the bearings under different conditions for each bearing. The dataset includes samples of different rotational speeds (900 rpm and 1500 rpm), different radial forces (400 N and 1,000 N), and different loads (0.1 Nm and 0.7 Nm). The data is from a test rig consisting of a test motor, a bearing module, a measuring shaft, a flywheel, and a load motor. Please refer to the descriptions of Lessmeier et al. [36] for more information.

Other datasets are also given in the literature, such as, for instance, a train bogie test rig [138] or a gearbox dataset [139]. However, these are rarely used and often not publicly available.

In general, most of the available datasets are laboratory datasets that exist out of bearings with artificially introduced faults. However, laboratory datasets have two drawbacks. The first is that they often have only single-point faults, and the second one is that the recorded data lacks the noise of other machine components (e.g., electric motors and cooling systems).

In addition to the publicly available datasets used in various research papers, this thesis exclusively uses a private dataset of Junker Maschinenfabrik GmbH. In contrast to the aforementioned laboratory datasets, all faults are genuine. They are the results of normal wear-out of the spindle and can be single-point faults as well as distributed faults. The recordings of each spindle have been carried out under three different rotational speeds (from 10% to 100% of the maximum rotational speed). The data was collected with accelerometers attached to the spindle in the x-axis and y-axis orientations. The spindle is designed for a maximum power of 4 kW and a maximum rotational speed of 23,000 rpm. The bearings of the dataset can be assigned to the same three different health conditions as the CWRU and the University of Paderborn datasets, which are inner ring fault, outer ring fault, and healthy bearing. In sum, there are 48 different measurements.

3.3.2.3 Supervised Learning with CNNs

The easiest way to use the datasets described above is to use them for supervised learning without transfer learning. There are solutions using the raw input data and solutions that use abstractions through different feature extraction methods as input for a CNN. Examples of solutions using the raw sensor data are found in Wen et al. [26] and Ince et al. [140]. Both used the raw sensor data but in different ways. Wen et al. used a 2D CNN to plot the raw sensor data stream of the CWRU dataset and split them into smaller parts. Using this approach, they achieved accuracies of up to 100%. By contrast, Ince et al. directly used the raw sensor data of a real-world motor conditioning system as input for a 1D CNN. This approach achieved a lesser accuracy of up to 97.8%. However, this approach has the benefit of faster training due to the fact that a 1D CNN has fewer trainable parameters than a 2D CNN. The work of Ding and He [125] used a feature extraction technique. They used wavelet packet energy (WPE) to convert the raw vibration data of the CWRU dataset into images. Therefore, the energy of a wavelet packet transform, a particular wavelet transform that uses high and low passes during the transformation, was plotted over time. The researchers reached an accuracy of 96.8% using this approach. Verstraete et al. [141] had a similar approach. They also used the CWRU dataset and compared different time-frequency domain transformations (HHT, Wavelet, and STFT) as well as different resolutions of the image (32x32 pixels and 96x96 pixels). The architecture of their approach was based on a double convolutional layer layout, which leads to an increased significance of the features through additional nonlinearity. A nonlinearity decision function is important for making complex nonlinear decisions [142]. The best approach reached an accuracy of 99.9%.

Jing et al. [143] investigated the influence of different feature extraction methods for different machine learning approaches like CNN, SVM, and a fully connected neural network. They showed that for their dataset, which was provided by the 2009 PHM Data Analysis Competition, CNNs perform much better than other machine learning approaches when using the input data directly due to their ability to learn features. The results were nearly equal between the different techniques by manually preselecting features like the crest factor. The best accuracy (99.33%) was obtained using a CNN with direct frequency domain data. This result is even better than the results achieved using manually selected frequency domain features. Although the researchers recommended the use of 2D segments for future work, they trained the CNN with 1D segments from raw data input.

An improvement over pure CNN approaches can be achieved by combining them with other techniques. This was done by You et al. [144], who introduced a hybrid model that uses a CNN together with an SVR. The CNN was used for extract features directly from the raw sensor data of the CWRU dataset. However, the SVR was used for classification. By combining the two techniques, they had a 97.6% accuracy rate for a larger dataset with different fault sizes and 93.9% for a dataset with only one fault size. These results were better than when using only one of these techniques.

The presented research shows that in the area of supervised learning, very high accuracies can be achieved (up to 99.99%). However, it should be noted that these high accuracies might be based on the fact that supervised classifications, in general, provide higher accuracies based on large labeled datasets [145].

3.3.2.4 Transfer Learning for Same Bearing Types with CNNs

As an extension to supervised learning, there are also solutions that use transfer learning of the same bearing type. In doing so, knowledge from datasets of the same type, which work under different process parameters (e.g., loads) or have different fault sizes, is transferred. All of the in the following presented works use the CWRU dataset. This dataset has the advantage of having, amongst others, samples of different fault sizes as well as samples of different loads, which can be used for transfer learning.

Li et al. [137] suggested a transfer learning approach for 1D sensor input data. Their proposed CNN was based on labeled data for the source domain and unlabeled data for the target domain. Both datasets were of the same bearing type, but each domain used different loads. During the training phase, they used data from the source and the target domain and adopted the two domains with the help of an MMD loss function. They evaluated combinations of different loads in the source and target domain as well as different MMD setups, such as classical MMD and MK-MMD. The results showed that the accuracy of each implementation is dependent on the transfer task. In all evaluations, they got results of about 95% accuracy with transfer learning. Without transfer learning, the accuracies were between 63% and 82%.

Zhang et al. [136] used a particular layout of a CNN where the first layer had a wide kernel. They also called this approach a wide deep convolutional neural network (WDCNN). They transferred knowledge between domains of different loads but with the same bearing type. The transfer itself was done by statistical information of the target domain. Unfortunately, the exact nature of this information is not specified. With this network, they reached an accuracy of 95.9% on average.

Li et al. [138] employed a train bogie test rig dataset and the CRWU dataset to a two-step transfer learning approach. In order to extract frequency-domain data, they applied an FFT on the raw data. In the following step, they used a CNN and a generative neural network to generate a high-level feature representation. The CNN extracted the features, while the generative neural network used the output of this CNN to generate fake fault data. To reduce the distribution discrepancy between the fake and actual features, they employed the variance and the mean of the target domain data. The trained network was also used for the target domain. In the last step, they utilized another CNN for the classification with an accuracy of above 92% in the target domain.

Han et al. [139] introduced an intelligent fault diagnosis framework called deep transfer network (DTN). Again, the source and the target domain datasets were from the same test environment. The difference in their approach is that they used different fault sizes. Their suggested framework uses Joint Distribution Adaptation (JDA) to adapt the conditional distribution of unlabeled target data together with labeled source domain data. JDA combines the classical MMD and an MMD that is calculated with the conditional distribution of each category as an input. With the help of JDA, they achieved an accuracy of 99.3% for transferring knowledge from small fault sizes to large fault sizes and 97.1% for the reverse direction.

All the presented transfer learning approaches for the same bearing type with different process conditions show promising results that are up to 99.3%. This shows that there is little reason to perform further investigations in this direction. However, it must be noted that all these works were performed on laboratory datasets, which have hardly any noise.

3.3.2.5 Transfer Learning for Different Bearing Types with CNNs

The aforementioned works may not perform particularly well when trained on a dataset of one bearing type and used to classify another. Therefore, there are also approaches for transfer learning for the classification of bearing faults between different bearing types. Since there are only a few solutions for transfer learning of different bearings based on accelerometer data, this chapter focuses on all kinds of machine learning techniques and not only on CNNs. In the following section, all, to the author's best knowledge, existing transfer learning approaches for different bearings based on accelerometer values are presented.

Yang et al. [146] suggested a feature-based transfer neural network (FTNN) to determine the condition of bearings in actual machines using the diagnosis knowledge from bearings in lab machines. To extract transferable features from both actual and lab machines, their suggested framework used a CNN. The parameters of the CNN were then constrained by regularization terms of a multi-layer domain adaptation (MMD-based) and pseudo-label learning to reduce the among-class distance of the learned transferable features. They used two bearing fault cases to validate the suggested approach. In both cases, the target domain data contained the health status of locomotive bearings. This dataset was a real-world dataset, and both source domains were laboratory domains. Case 1 contained data from motor bearings, and the second case contained gearbox bearing data. The results showed that the suggested approach can successfully learn transferable features to link the discrepancy between the data from actual and lab machines by having an accuracy of 84.32% for case 1 and 74.81% for case 2.

Zhiyi et al. [147] presented an improved deep transfer auto-encoder for the purpose of diagnosing faults of bearings placed across multiple machines. An auto-encoder is a type of ANN that is typically used to learn a condensed representation (encoding) of the input data. A scaled exponential linear unit was employed in their suggested framework to increase the quality of the mapped vibration data acquired from bearings. Furthermore, a nonnegative constraint was used to change the loss function in order to enhance the reconstruction effect. The trained source model's parameter knowledge was then transferred to the target model. To harmonize the properties of the target test samples, target training samples that had limited labeled information were utilized to fine-tune the target model. The vibration signals of bearings installed in multiple machines were used to evaluate the suggested model. The researchers verified their approach in two case studies: Case study 1 used the CWRU dataset as the source domain and a private test bench dataset as the target domain. An accuracy of 90.42% was achieved. Case study 2 used a gearbox dataset as the source domain and a dataset of bearings of a locomotive wheel as the target domain. Here, an accuracy of 88.18% was reached. Only the target domain data of case 2 was a real-world example. This dataset was very small, and it is not apparent if samples from the same bearing were used in both the training and test datasets. Overall, the presented results indicate a promising performance compared to existing techniques.

Cheng et al. [83] suggested a deep transfer learning approach that uses relevant information from the source domain to perform learning in the target domain. Therefore, they used the Wasserstein distance to minimize the distributions between the source and target domain. In their setting the Wasserstein-based transfer learning approach has shown to perform well for both unsupervised and supervised learning. They used the CWRU dataset for different transfer tasks: unsupervised transfer learning of different speeds (average accuracy 95.75%), unsupervised learning of different loads (average accuracy 64.20%), and supervised transfer learning of different locations (average accuracy 64.92%). Different locations implicitly lead to different bearings since different bearings are mounted in both locations of the CWRU dataset. The difference in accuracies can be explained by the fact that the speed difference is only minor; therefore, this transfer task might be easier.

To summarize, there are only three solutions for transfer learning between different bearing types. They are based on different algorithms and two of them target a real-world dataset. Independent of the algorithm, the accuracy is between 65% and 90%.

3.3.2.6 Conclusion

The state of the art for bearing fault classification has been presented in the previous sections. Independent of the setup (supervised learning or transfer learning), one of the challenges of all researchers is that only a few datasets are available to validate the different approaches. The most frequently used dataset is that of the CWRU. This dataset and most of the other datasets are laboratory datasets that exist out of bearings with artificially introduced faults. In addition, there is no or only little noise from other components.

Table 6: Brief overview of different research in the field of bearing fault classification.

	Authors	**ML model**	**Signal Processing**	**Transfer function**	**Difference between source and target data**	**Different process conditions in the target domain**	**Accura-cy**
Without Transfer Learning	Wen et al. [26]	CNN	Raw data	-	-	No	100%
	Ince et al. [140]	CNN	Raw data	-	-	No	97.8%
	Ding and He [125]	CNN	Wavelet packet energy	-	-	No	96.8%
	Verstraete et al. [141]	CNN	HHT, Wavelet, STFT	-	-	No	99.9%
	Jing et al. [143]	1D CNN	Frequency data	-	-	No	99.33%
	You et al. [144]	CNN +SVR	Raw data	-	-	No	93.9% - 97.6%
Transfer learning with same bearing types	Li et al. [137]	1D CNN	Raw data	MMD	Different operating loads	No	95%
	Zhang et al. [136]	CNN	Raw data	Mean and Variance of the target domain	Different operating loads	No	95.9%
	Li et al. [138]	CNN, GNN	FFT	MMD	Different fault sizes	No	92%
	Han et al. [139]	DTN	Raw data	MDA, JDA	Different fault sizes and operating loads	No	97.1% - 99.3%
Transfer learning with different bearing types	Yang et al. [146]	FTNN	Raw data	Multi-layer MMD	Source from Lab Target from real machines	No	74.81% - 84.32%
	Zhiyi et al. [147]	auto encoder	Raw data	Custom	Source and target from different machines	No	88.18 - 90.42%
	Cheng et al. [83]	DTL	FFT	Wasserstein	Different setups with different bearing types, operating loads, and rotational speeds	No	64.20% - 95.75%

The datasets are used in a supervised manner or completely unsupervised for transfer learning. Unfortunately, there are no solutions that use partly labeled datasets for semi-supervised training. As stated in the problem statement, semi-supervised learning is of great interest since it is difficult for many companies to have all data labeled. Many enterprises are already happy to collect data and disassemble some of the defective bearings.

There are many approaches for the classification of bearing faults for fully labeled datasets. As shown in Table 6, some solutions have an impressive success rate of 99% [143] or even more [26]. These solutions, however, are designed for one particular use case, usually with a certain type of bearing, fault size, and load. There are also transfer learning solutions for different process parameters and conditions of the same bearing, which can accurately classify the condition of a bearing. This changes when the transfer between different bearing types is considered. Here, the accuracy drops to between 65% and 90%. This clearly shows that this transfer task between bearings is more sophisticated. Table 6, which gives a brief overview of the presented research, shows that until now, no approach has combined the transfer learning fields of different process parameters in the target domain and different bearings in the source and target domain. Therefore, one missing use case is to have a partly unlabeled target domain with different rotating speeds of the spindle/bearing. This use case is also a real-world use case and can be related to different machine manufacturing processes. From this deficit, the following first research question of this thesis can be derived directly.

RQ1: What are the necessary characteristics of a new classification method for bearings, which can take the benefits of a dataset of a different bearing type for a partly labeled target dataset that is collected under different process conditions?

3.3.3 Remaining Useful Life of Bearings

3.3.3.1 Introduction

Fault classification is an essential aspect of condition monitoring. However, knowing the remaining useful life is also important to prepare planned maintenance and avoid a machine breakdown. This can be accomplished by reporting the current condition, such as “new,” a percentage of the lifetime, or even an estimate in operational hours. In general, there are two common approaches for estimating the RUL. The first is to employ a health indicator, which is often a value between 0.0 and 1.0, with 0.0 representing a healthy condition and 1.0 representing a defect. With the help of this indicator and the total lifetime of samples, a mapping from the health indicator to the remaining time can be made [148]. The other approach is based on regression. Here, it is possible to work directly with the remaining time. For this purpose, the future trend is determined based on past values. The RUL can then be calculated directly using this trend [12, 149]. SVMs [149] and RNNs [148] are two well-known approaches for this. Markov Models (MM) [150], Mahalanobis Taguchi systems (MTS) [151], and other artificial neural networks [152] are other used approaches. Furthermore, there also exist concepts that are not depending on machine learning, such as the closed skew s-normal (CSN) distribution [153]. The CSN is determined at each data point by taking the RMS of the vibration data. A regression model can also be used for traditional approaches, as done by Zhao et al. [53]. As described in Sections 2.2.1 and 3.2.6, machine learning approaches are more suitable for complex and dynamic datasets and do not need manual fine-tuning. Therefore, non-machine learning approaches are not presented here and are beyond the scope of this thesis.

Most studies estimating the RUL do not use the raw data directly. Instead, they use calculated features of the time or the frequency domain, such as crest factor or root mean square [54]. The datasets consist mainly of one dataset that is described in the following section. Afterward, some relevant works that use supervised learning without transfer learning are reviewed. This is followed by a review of relevant works for transfer learning for the same bearing type. Finally, to the best of the authors' knowledge, only two existing transfer learning approaches for different bearing types are introduced.

3.3.3.2 Reference Datasets

There is one publicly available dataset that is used in most publications: that of the FEMTO Institute, which was used at the IEEE PHM 2012 data challenge competition [120]. This data is acquired from an accelerated aging platform called PRONOSTIA, which provides vibration and temperature measurements. The vibration signal is measured by two high-frequency accelerometers (type 3035B DYTRAN), which provide data with a sampling frequency of 25.6 kHz. One is used horizontally, and the other in a vertical direction. Every 10 seconds, 2560 data points are recorded. The data of each bearing was recorded until the vibration signal exceeded 20 g. This condition was then considered as defective. The defects of all bearings, which can be single-point or multiple-point, are based on natural degradation. There are no seeded faults. There are three groups of different operation conditions: the first (1800 rpm and 4000 N) and the second (1650 rpm and 4200 N) contain seven bearings each, while the third (1500 rpm and 5000 N) contains only three bearings.

3.3.3.3 Supervised Learning with RNNs

As mentioned in Section 3.3.3.1, the two important machine learning techniques used for supervised learning in the context of RUL are SVMs and RNNs. In the case of SVMs, there are two relevant research papers. The first is from Liu et al. [154], who proposed an approach for RUL based on numerous health state assessments. Their approach split the entire bearing life into different health states. The authors built an individual regression model with unsupervised and supervised components for each state. In a first step, the unsupervised part used principal component analysis (PCA) and clustering to automatically extract knowledge in the form of health state labels using a fuzzy c-means technique. PCA is a technique to restructure a dataset by approximating a large number of statistical variables using a few linear combinations. An SVM was used afterward to detect the health status using the generated labels. With the help of these generated labels, an SVM was then used to detect the health states. Afterward, the RUL was estimated with another SVM. Finally, the authors evaluated their approach based on a custom benchmark that compared the difference between the true and the estimated absolute RUL of two bearings: bearing 1_3 (true RUL: 5730, estimated 5842) and bearing 1_4 (true RUL: 339, estimated RUL 1109).

The second paper contains the winning approach of the IEEE PHM 2012 data challenge competition [120]. Sturisno et al. [155] compared three different approaches. The first was a Bayesian Monte Carlo approach, which was based on an exponential model. This approach did not perform very well because it could not handle the abrupt signal changes at the end of life. The second was an SVR-based approach, which worked better, but more datasets would have been needed to increase the accuracy. The third approach was based on detecting anomalies by just examining variations in the frequency signature. Based on these changes, the authors calculated an anomaly ratio that was used to estimate the RUL based on an equation. In terms of the prediction accuracy, the third method was the most precise one. By using the challenge benchmark setup, the authors reached a PHM score of 0.306. For a detailed explanation of the calculation of this score, see Appendix A.5.1.

There are also approaches based on deep learning, such as RNNs. Guo et al. [148] proposed an RNN-based health indicator for estimating the RUL of bearings. They chose 14 features from the time, frequency, and time-frequency domains. The features of the time and frequency domain were transformed to related similarity features. Thereby, the current state was compared with the previous states. By using monotonicity and the correlation between operating time and features for training of their RNN, the authors chose the best features. Their approach was the best in their benchmark, with a mean relative error (*Er*) of 23.24. For the calculation of *Er,* see Appendix A.5.1.

Malhi et al. [156] utilized an RNN and their own test data, which was captured at 1200 rpm. They employed ten features: five time-domain features (crest factor, kurtosis, peak value, rectified screw, and RMS) computed from raw data, and the same five features estimated from the result of a Morlet mother wavelet-based continuous wavelet transform. The authors trained their neural network using all data up to a certain point. Then, they forecasted the upcoming degradation trend. This is a real-time recurrent learning approach that allows the RNN to be further trained while in use. As a result, on a machine, all currently available data can be utilized to estimate the future trend. This approach showed the best results in their benchmark based on a mean squared error (MSE) of 0.05.

Zhang et al. [157] used a two-layered network. In the first layer, they used a CNN, which is followed by an LSTM. The raw sensor data was used as input for this network. In addition, this data was used to calculate a health indicator based on 23 features of the time, frequency, and time-frequency domain. The authors used IEEE PHM 2012 data as well as the challenge's setup and scoring function. The result was a PHM score of 0.64.

All the presented works seem to be successful. However, although they all (with the exception of Malhi et al. [156]) use the dataset of the IEEE PHM 2012 data challenge, they are not comparable because they use different RUL mechanisms as well as different criteria for benchmarking their approach.

3.3.3.4 Transfer Learning for Same Bearing Types with RNNs

The dataset of the IEEE PHM 2012 data challenge is not only used for supervised learning. One of the relevant research approaches in the context of transfer learning also uses this dataset. This is done by Cheng et al. [12], who used a CNN for transfer learning the knowledge of one fault behavior to another of the same bearing type. The authors split the dataset into groups of different fault behaviors. To learn domain invariant features of the raw sensor signal they used MK-MMD. Their approach obtained the highest accuracy in their benchmark with methods without transfer learning based on a mean absolute *Er* of 47.35.

Another important work is that of Liu and Gryllias [82], which is based on an XJTU-SY bearing dataset. The authors also used transfer learning to adapt to different working conditions of the same bearing type. Their approach was based on combining a domain adversarial neural network (DANN) and a bidirectional LSTM neural network (Bi-LSTM). On the one hand, the DANN was used to deal with domain shift (e.g., different distributions of the datasets). This mechanism is inspired by GANs and tries to generate domain invariant features for the source and the target domain data. On the other hand, the Bi-LSTM was used to extract the features for estimating the RUL. As already mentioned in Section 2.4.5, the usage of a Bi-LSTM leads to higher accuracies than only using an LSTM because the training of a Bi-LSTM also takes the states of future timestamps into account. The authors benchmarked their result based on the root mean square error (RMSE), which was between 11.52 and 20.39, and the mean absolute error (MAE), which was between 4.21 and 16.54.

3.3.3.5 Transfer Learning for Different Bearing Types with RNNs

In addition to the previously introduced transfer learning approaches for the same bearing type, there are also solutions for transfer learning between different bearing types. However, to the best of the author's knowledge, only two existing studies use transfer learning. Both are discussed in this section.

Like most other RUL approaches, the work of Xia et al. [158] used the dataset of the 2012 IEEE PHM Data Challenge as the target domain dataset, whereas the source domain dataset was from Case Western Reserve University [121]. Raw sensor signals were used as input of this approach, which exists of two parts: a fault knowledge transfer neural network (FTNN) and a convolutional LSTM ensemble network. The FTNN was made of three consecutive pairs of one convolutional layer and one pooling layer. The target domain dataset was used to train this neural network initially. The transfer learning process began with the pre-trained but unfixed convolutional layers. Then, this network was trained with inputs from both domains (source and target) at the same time. This was accomplished through the usage of MMD. The trained FTNN's output was then fed into the LSTM ensemble network. This network consisted of *n* parallel LSTM networks with a similar layout. Each of the *n* networks was designed to handle one bearing condition. Finally, the estimated RUL was calculated using an ensemble mechanism that weighted the outputs of the LSTMs. To validate their approach, the authors used their own validation setup, which utilized one working condition out of the three available in the dataset. They also increased the number of training samples by switching the learning test ratio of the datasets of the challenge from two train and five test datasets to five train and two test datasets. The results were determined by the RMSE, showing an RMSE of 0.0673 and 0.1631, which leads in a reduction of up to 48.61% when compared to other self-implemented approaches.

Huang et al. [159] suggested an analogous approach for transfer learning between different bearing types. The source domain dataset was again the 2012 IEEE PHM Data Challenge dataset, and the target domain was from the Intelligent Maintenance Systems (IMS) [160]. They passed the raw sensor input into convolutional and pooling layers. Their output was fed into a bidirectional LSTM. The result was then used as input for fully connected layers to estimate the RUL. Instead of the commonly used Adam optimizer, they employed an adaptive hybrid high-power multi-dimensional gradient algorithm (AHHPMG), which is their own backpropagation algorithm. AHHPMG takes into account the temporal correlation of the measurements in the training data. First, they pre-trained the network with the RUL dataset of the source domain. Afterward, the pre-trained network was trained using the target domain data in the same manner. The target dataset was not divided based on bearing instance. This means that the training dataset contained many samples of the same bearings, which were also used for testing. In the author's opinion, this method is invalid because all it does is a complex interpolation of the RUL values in the training samples. The researchers validated their results using two bearings based on the normalized root mean square error (NRMSE) (bearing 1: 0.497, bearing 2: 0.234) and the mean absolute percentage error (MAPE) (bearing 1: 16.311, bearing 2: 23.124).

In conclusion, both approaches use the 2012 IEEE PHM Data Challenge dataset. Furthermore, both use the entire frequency range of the data being recorded with a high-resolution sensor. In addition, both approaches make use of their own evaluation metric. Consequently, the outcomes are not comparable with most other approaches, which often use the challenge setup and the PHM score (see Appendix A.5.1) for their evaluation.

3.3.3.6 Conclusion

To summarize, as shown in Table 7, there are various approaches to RUL without transfer learning based on SVM, CNN, RRN, and LSTMs. This table also shows that there are only two transfer learning approaches for RUL of bearings of different bearing types. Although the input of all approaches is based on only a few datasets, they are not directly comparable because they often use custom benchmark setups as well as different metrics to measure the performance of their approach. A benchmark type that is often used and well-defined is the IEEE PHM 2012 challenge setup and the PHM score (see Appendix A.5.1).

All presented approaches use the entire frequency range of data that is collected with costly accelerometers with high sampling rates as input. However, the permanent mounting of expensive sensors does not reflect the industry's real needs where solutions are needed with a good return on investment. This need can be fulfilled by using triaxial sensors with low sampling rates (see Section 3.2.5). This use case has not been researched yet and represents a clear research gap regarding suitable solutions for transfer learning of different bearing types that also cover the usage of sensor data with low sampling rates (up to 5000 Hz).

This shortage leads to the second research question of this thesis:

RQ2: What are the necessary characteristics of a new RUL method for bearings, which can take the benefits of a dataset of a different bearing type, for a labeled target dataset that is recorded with sensors with low sampling rates?

Table 7: Brief overview of different research in the field of transfer learning for estimating the RUL.

	Authors	**ML model**	**Signal Processing**	**Difference in source and target data**	**Transfer function**	**High-frequency data**	**Performance**
Without Transfer Learning	Liu et al. [154]	SVM	PCA	-	-	Yes	Bearing 1_3: True RUL 5730, estimated 5842 Bearing 1_4: True RUL 339, estimated 1109
	Sturisno et al. [155]	Mathe-matical equation	FFT	-	-	Yes	PHM score 0.306
	Guo et al. [148]	RNN	14 features of the time, frequency, and time-frequency domains	-	-	Yes	Mean Er 23.24
	Malhi et al. [156]	RNN	Five time domain features	-	-	Yes	MSE: 0.05
	Zhang et al. [157]	CNN + LSTM	Raw data; HI based on 23 features	-	-	Yes	PHM score: 0.64
Transfer learning same bearings	Cheng et al. [12]	CNN	FFT	Different fault behaviors	MK-MMD	Yes	Mean Er: 47.35
	Liu and Gryllias [82]	Bi-LSTM	Raw data	Different working conditions	Adversarial Training	Yes	RMSE 11.52 – 20.39 MAE 4.21 – 16.54
Transfer learning with different bearings	Xia et al. [158]	LSTM	Convolutional and pooling layers	Different bearing types	MMD	Yes	RMSE: Bearing 3: 0.0673 Bearing 5: RMSE 0.1631
	Huang et al. [159]	Bi-LSTM	Convolutional and pooling layers	Different bearing types	Pre-trained model	Yes	NRMSE: Bearing 2_1: 0.497, Bearing 3_1: 0.234 MAPE Bearing 2_1: 16.311, Bearing 3_1: 23.124

3.3.4 Conclusion

The state-of-the-art review for predictive maintenance solutions for bearings shows various activities in both the classification of bearing faults and remaining useful life. However, both research areas still have unaddressed aspects, which are mainly based on the lack of proper datasets. One of them is the possibility of using datasets of different bearings in the source and the target domain in combination with different process parameters. Out of this requirement, RQ1 for the task of bearing classification has emerged. The second is a solution that covers the need to use low costs sensors in combination with transfer learning. RQ2 covers this need for the use case of RUL estimation.

The state of the art has shown that independent of the RUL or the classification task, aside from the ML model used, two other processes are also important: the signal processing and the transfer learning method. In the case of unsupervised transfer learning, the most promising results are achieved using discrepancy-based loss functions like MMD. In addition, CORAL is also a successful transfer learning loss function.

In the case of the signal processing method, the literature has shown that there are many different ways to use the sensor data (e.g., through time-frequency transformations). These methods are necessary for the ML algorithm to have usable features. However, they are not specialized for the use case of using different domains. Some of the feature extraction methods can be used without modifications on different bearings (e.g., FFT), and some have to be adapted for each dataset (e.g., wavelets). The aforementioned lack of training data for predictive maintenance task as well as the lack of IT specialists, especially in Small and Medium-size Enterprises (SMEs) [161], lead to the following question, which is also defined as RQ3:

RQ3: What are the necessary characteristics of a feature extraction method that is well suited for transfer learning? This method must be stable enough to be used on different bearing types without changing its parametrization or making significant changes to a subsequent machine learning model for different bearing types.

3.4 Conclusion

This chapter showed that predictive maintenance is a very relevant topic. Large industrial plants and companies pay particular attention to Industry 4.0 and, inevitably, to predictive maintenance. The presented use cases for transfer learning for predictive maintenance tasks of bearings are relevant for many manufacturing plants. A lot of different types of machines use spindles with bearings inside. Since the individual machines are so different, transfer learning is the only solution to achieve accurate results since there is too little data for traditional ML approaches.

The relevance of bearing faults classification and estimation of remaining useful life is also evident from the number of scientific publications on these topics, as presented in Section 3.3. As such, a particular focus was given to the methods used and their constraints, as reflected by RC1 and RC2. Based on the limitation of these methods, the research questions of this thesis were derived. This thesis will present appropriate solutions for these questions in the following chapters. The next chapter (Chapter 4) presents an intermediate domain, which can be seen as an answer to RQ3. This is followed by a solution for the classification challenge in the form of a general approach for different bearing types and different process conditions in the target domain (Chapter 5). This solution can also be seen as the answer to RQ1. Chapter 6 presents a solution for the RUL challenge of transferring knowledge from one bearing type to another. This solution is tailored for the need to use samples of sensors with low sampling rates and is also part of the answer for RQ2.

4 Feature Extraction

4.1 Introduction

The previous chapters, 2 and 3, have shown the current state of the art for predictive maintenance solutions in general and for the tasks of classification of bearing defects and RUL estimation of bearing in particular. Independent of the task, the beginning of the processing chain for a predictive maintenance process is the same: It always starts with raw sensor data. As shown in Figure 22 and mentioned theoretically in Section 2.3, there are different ways to use this data. It can be used directly or after a feature extraction has been performed. There are feature extractions available that are based on a transformation into a different domain, such as frequency (with an FFT) or time-frequency-domain (with an S-transform, HHT, or STFT). In addition, statistical features such as the shape factor can be applied to the data. If data from different domains is used, there is also the possibility of using an intermediate domain, which combines the features of both domains.

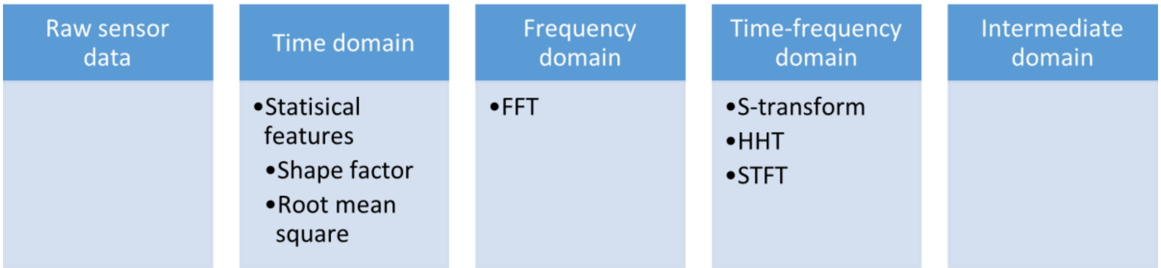

Figure 22: Different feature extraction methods for sensor values in a machine learning process: Direct raw sensor data, statistical features, such as shape factor in the time domain, transformation into different frequency and time-frequency domains, and the use of an intermediate domain.

One target of predictive maintenance applications is components that make periodic movements. These provide patterns with distinctive characteristics in the frequency range that depend on the periodicity. The in Section 2.3 presented feature extractions are similar in that they do not take care of the context information based on these patterns. In use cases with only small datasets, context information might help to increase the accuracy of the machine-learning algorithm (see Section 2.4.2). In addition, this context information can also help to improve the transfer learning between different domains/datasets. Another downside of some of these methods is that they must be adjusted for each application. Therefore, a feature extraction method is of interest that fulfills the following requirements:

- It must be a feature extraction method that takes care of the patterns provided by components with periodic movements.
- It should increase the predictive maintenance quality of such components based on frequency-related patterns.
- It should improve transfer learning between different component types.

- It should be a stable solution that can be used on different types of a component without modifications.

A feature extraction method that meets the above-listed requirements is also important for the use case of bearings. Therefore, this chapter focuses on a solution in the context of bearings. This also addresses RQ3, which asks for a feature extraction method that is well-suited for transfer learning. Furthermore, this method has to be stable enough to be used on different bearing types without changing its parametrization or significant changes on a subsequent machine learning model for a different bearing type.

A possible solution for this RQ might be to use a specialized feature extraction type which is called an intermediate domain. In this case, an intermediate domain can use the input data and modify it in a way that only the relevant features of the sensor signal are in focus. This feature selection could lead to benefits in accuracy (see Section 2.3.1). In addition, an intermediate domain could lead to better results for transfer learning use cases, as it is itself a type of transfer learning that brings the source and target domain closer together (see Section 2.4.3.3).

Therefore, this chapter presents an intermediate domain-based feature extraction method for raw sensor data of accelerometers connected to bearings. As a first step, the validation context for this chapter is introduced (Section 4.2). This is followed by a detailed justification for the decision to use an intermediate domain (Section 4.3). The proposed intermediate domain itself is created through three sequential steps (see Figure 23), which are explained as follows: The first step is the processing of the raw sensor data to convert it into the time-frequency domain for better analysis possibilities (Section 4.4). This is followed by a de-noising algorithm to obtain a stable intermediate domain for different bearings and process conditions independent of noise (Section 4.5). Finally, the output must be prepared for use in classification or an RUL task (Section 4.6).

Each of these subchapters first introduces the reason for choosing this step and then explains it in detail.

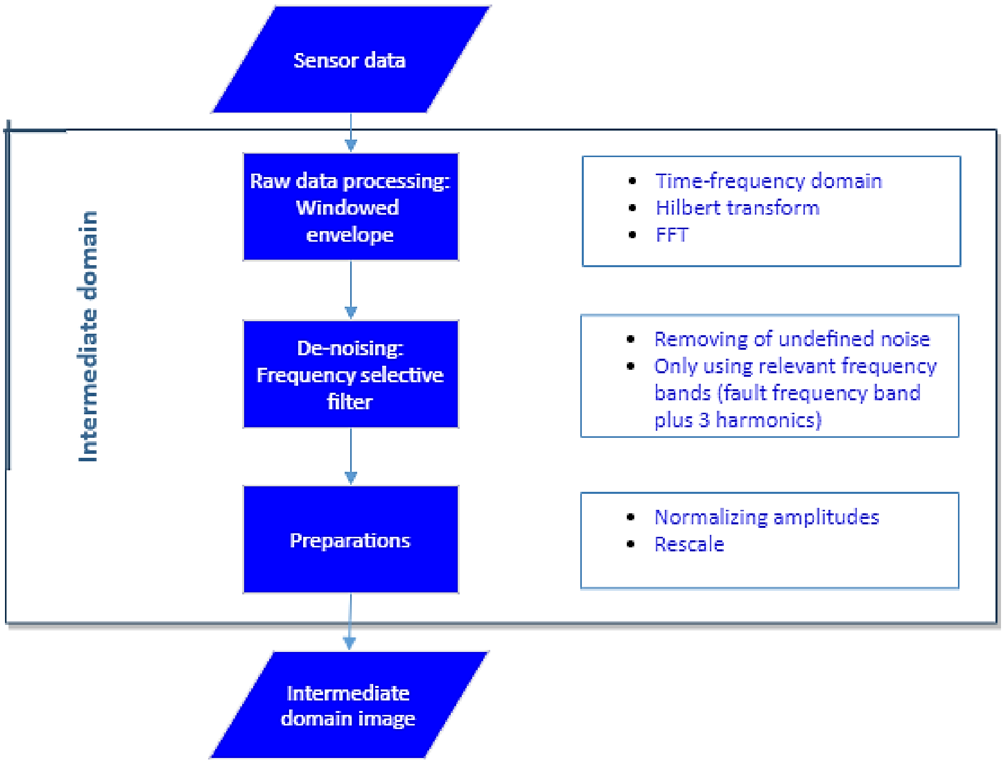
Sensor data
Intermediate domain
Raw data processing: Windowed envelope
Time-frequency domain
Hilbert transform
FFT
De-noising: Frequency selective filter
Removing of undefined noise
Only using relevant frequency bands (fault frequency band plus 3 harmonics)
Preparations
Normalizing amplitudes
Rescale
Intermediate domain image

Figure 23: The creation of an intermediate domain image. As a first step, the raw sensor data is converted into the time-frequency domain to obtain better analysis possibilities. Afterward, the transformed data is de-noised with a frequency-selective filter. Before the created image can be used, it must be prepared for the upcoming machine learning algorithm (cf. [133]).

4.2 Validation Context

In order to validate the decisions of the different development steps of the intermediate domain, different bearing health-based test scenarios have been used. Each test scenario uses the CNN presented in Section 5.4.2 for the classification of bearing faults. For the test scenarios that are used for the raw signal processing presented in Section 4.4, the data provided by the Case Western Reserve University (see Section 3.3.2.2) is used. This laboratory dataset without noise and no different process conditions has been chosen because the test scenarios of this chapter should only focus on signal processing and not noise or process conditions. On the other hand, Section 4.5 deals with filtering out the noise and the usability under different process conditions. Both properties are given with the custom dataset of Junker Maschinenfabrik GmbH (see Section 3.3.2.2), which is why it was selected for this chapter. Independent of the used dataset, the process is always the same: For each test scenario, the dataset is split bearing instance based on a ratio of 70% training data and 30% test data. The result of one test scenario is the mean of two runs with a different training and test data split. This split is identical between all test cases in one test scenario and is based on a random assignment. For each training and test run, the datasets are similarly assigned to training and test data. The detailed assignment of the data sets can be seen in Appendix A.1.1 in Table 29. The parameters of the training itself are listed in Table 8.

Table 8: Parameters used during the training of the neural network.

Parameter	Value	Reason
Optimizer	Adam	Empirical tests and recommendations such as [6]
Learning rate	0.00003	Empirical tests
Batch size	150	Empirical tests
Iterations	200	Empirical tests
Loss function	Cross-entropy	See Section 5.4.2

4.3 Intermediate Domain Structure

As introduced in Section 4.1, there are different methods for feature extraction of sensor data. The recommendation for using a feature extraction method instead of the raw sensor data is given by various studies in different domains, such as that of Sadouk [162], who compared time-frequency domain feature extraction (in the form of an S-transform) to raw data for datasets of different domains (like human activities) using CNNs. His finding was that feature extraction leads to an improvement in accuracy compared to the usage of raw data. This has also been shown for the specific use case of predictive maintenance tasks for bearings through the state-of-the-art survey in Section 3.3.

A particular type of feature extraction is the usage of an intermediate domain. As mentioned in Section 2.4.3.3, an intermediate domain based on feature representation can improve the accuracy of classical machine learning and transfer learning tasks. This is because the intermediate domain itself is already a transfer learning method. As introduced in Section 2.4.3.3, it is a transductive transfer learning method that is meant to reduce the discrepancy of the features of the input data between the source and the target domain to achieve better accuracies in classification and regression models. This is especially important when performing heterogeneous domain adaption [61] and has been verified, for instance, by Zhang et al. [163]. They used the intermedia domain for an image classification setup to benefit from the locality geometric structure of domain data. In most test cases, the intermediate domain achieved the best accuracy. Another advantage of using an intermediate domain is that domain knowledge can be incorporated here. Thereby, a hybrid approach can be used to bridge the gap between a purely data-driven and a model-based approach (see Section 2.2.2). The resulting hybrid approach can use the benefits of a model-based approach in a way that features, which clearly have no effect for the given task, can be removed through domain knowledge. This can decrease the training complexity, which is especially important when using only a small dataset.

The requirements of the feature extraction method include the real-world need to be a stable solution that can be reused without changing its parametrization. This means that the proposed intermediate domain should be used on different bearings and process conditions without modification. This results in the same intermediate domain being used to transfer the raw sensor data of a source domain and a target domain to an abstraction that can be used for both RUL and classification tasks. Therefore, the following applications are possible: As shown in Figure 24, periodic test runs can be used to capture the measuring data. Therefore, a component is tested after a test interval *T* for a measuring length *t*. Then, these measurements are converted into intermediate domain images. Out of these images, either only the last measurement can be used (e.g., for classification) or several can be used (e.g., for RUL). In addition, it is also possible to perform a manual test run primarily for classification tasks.

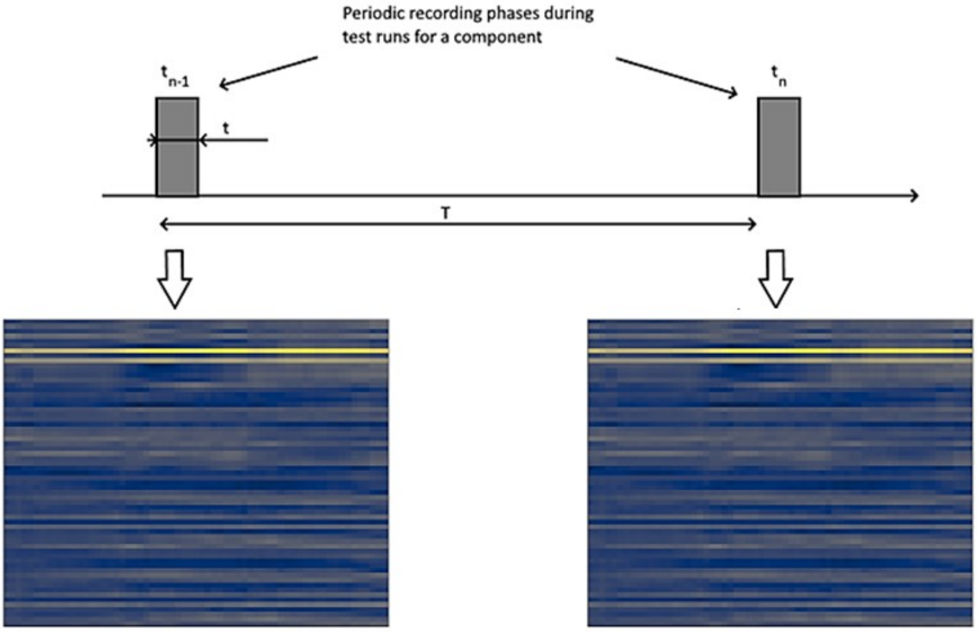

Figure 24: A setup of periodic test runs. After each test run, performed after an interval T, the sensor data measuring sequence of the length t cans be used for the given predictive maintenance task. The predictive maintenance tasks start with transforming the raw sensor data into an intermediate domain image.

4.4 Raw Signal Processing

4.4.1 Introduction

The Industry 4.0 development path describes the predictive maintenance, which is in development stage 5, relies on sensor data (development stage 3) (see Section 3.2.2). Before the sensor data can be used for a predictive maintenance use case, the domain in which the feature extraction should be applied to the data has to be selected. This is independent of the used sensor type, such as ultrasound, thermography, voltage, or dynamic pressure [1, 2].

While the measuring values of some sensors, such as temperature sensors, can be evaluated directly in the time domain, values of other sensors, such as sensors for detecting vibrations, should first be converted into the frequency domain or the time-frequency domain for better analysis [1, 162]. For each domain, a range of techniques is available, and the appropriate technique has to be selected. For instance, for the time domain, there are statistical features, such as the mean value (see Section 2.3.2).

If the data has to be analyzed in the frequency domain, one must decide if the frequency domain itself is sufficient or if it should be analyzed in the time-frequency domain. As discussed in Section 2.3.4.1, time-frequency techniques such as STFT or CWT must be used if the signal is non-stationary. Such signals appear when process parameters, such as speed, change during the measurement. If the signal is also nonlinear, an HHT or S-transform must be chosen. For simple stationary signals, both approaches can be used.

4.4.2 Windowed Envelope

Vibration analysis, which is often based on values of accelerometers [130], is used to monitor movements. Those movements can be rotations, which appear in components such as rotors, gears, and bearings. The common feature of rotating components is that they all have periodic signals, which are based on the rotation and can be analyzed in the frequency domain. Each component may have its fault pattern, which is based on its characteristic fault frequency, its harmonics, and its relation to other distinctive frequencies [39]. Several approaches can be used for the transformation in the frequency domain, like an FFT, as well as in the time-frequency domain, like an easy-to-implement STFT or more complex algorithms such as HHT or S-transform. As stated in Section 2.3.6, the latter can analyze non-stationary nonlinear signals at the cost of more complex and computationally intensive algorithms. However, since many predictive maintenance use cases can be assumed to have stationary and linear signals, algorithms such as STFT can also be considered. This is especially true for the scenario of using test runs, as described in Section 4.3. There is also a big advantage in using a simpler algorithm for real-world scenarios because predictive maintenance tasks are often performed by embedded systems that have low performance [1].

For the predictive maintenance task with non-machine learning based methods, the envelope analysis is widely used, since defects are more distinctly analyzable in the envelope spectrum (see Section 2.3.3.2). This is also evident in commercial products for bearings diagnoses that use this algorithm, such as those from IFM [130]. The envelope spectrum benefits from its ability to separate periodic signals from random noise by applying the Hilbert transform to the raw sensor signal. As described in Section 2.3.3.2, this transform is then followed by an FFT to extract periodically occurring frequencies. Since the envelope analysis is only a frequency domain operation, it may suffer from the aforementioned disadvantages of pure frequency transformations. An option to overcome these disadvantages may be to combine the envelope analyses with a time-frequency domain approach. This would allow the new technique to combine good detection of periodic signals with the ability to detect non-stationary signals. There are different time-frequency domain techniques available, which could be used as a basis for the novel approach. As stated above, STFT seems to be a promising technique for predictive maintenance tasks because it fulfills the requirement to be used for this task and its performance benefits compared to techniques like HHT and S-transform (see Section 2.3.6). As mentioned in Section 2.3.4.2, STFT is realized in three steps: First, the sensor data is split into small sections using a sliding window. This is followed by an FFT analysis, which is performed for each window. Finally, the results are concatenated in order to receive the time-frequency data of the signal. This makes this algorithm well-suited to integrate an envelope analysis. For the integration, the FFT analysis of the STFT can be replaced with an envelope analysis. All other parts, mainly the sliding window, can be taken over. Therefore, this new approach is called windowed envelope.

Different techniques have been benchmarked to select the best technique for the use case of bearings with their characteristic fault frequencies and to validate the performance of the windowed envelope. For this benchmark, the use case of bearing fault classification has been selected. The signals of the tested dataset are all recorded with a constant rotational speed, which leads to a stationary fault signal, which can also be analyzed in the frequency domain. For the verification, the raw sensor signals have been converted into the frequency domain using a classical envelope analysis and into the time-frequency domain using the here-presented windowed envelope as well as the two most sophisticated time-frequency domain approaches, HHT and S-transform. A detailed explanation of the verification setup can be found in Appendix A.2.1. The results, displayed in Figure 25, show that the windowed envelope has the best accuracy of all tested transformations for the tested dataset. In addition, the results reveal that a time-frequency conversion can have better results even on stationary signals. This indicates that the windowed envelope is a suitable technique for performing classifications of bearings in predictive maintenance tasks.

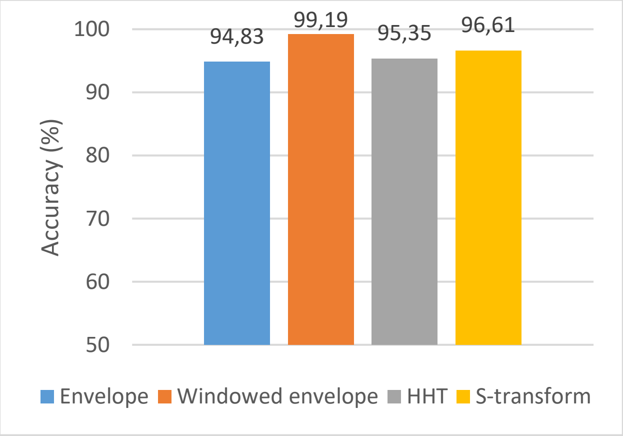

Figure 25: Classification accuracies of three different labels for bearings with different frequency and time-frequency conversions. The best accuracy is achieved with the windowed envelope. The test setup is described in Appendix A.2.1.

4.4.3 Signal Segmentation

A sensor provides continuous signals according to its sampling frequency. In order to use these signals as input for a transformation, like the one presented in the previous subchapter, the signal stream must be divided into segments of a specific length. As stated in Section 2.3.4.2, longer signal sequences, which cover more periods of a relevant frequency, are less vulnerable to the leakage effect. It is expected that the length of the segment, assuming that it is long enough, is not important.

Three different signal lengths have been chosen to validate this assumption: 0.11 seconds, which can cover ten times a complete 90 Hz signal; 0.2 seconds based on literature recommendations such as [164], and 0.7 seconds to cover the case of a longer time span. Those segment lengths have been used in a test scenario for the classification, which is based on the same setup as the test scenario for the different transformations (see also Appendix A.2.2). The results, which are presented in Figure 26, indicate that the accuracies are nearly independent of the signal length. Thus, the assumption made above is correct. In order to get a stable solution, as demanded in the requirements in Section 4.1, it is not sufficient to look only at the accuracies. It is also essential that the entire range of combinations of possible frequencies can be covered. Special attention must be paid to a process with low fault frequencies. For instance, there are processes that have fault frequencies that are lower than 85 Hz (see Section 3.2.4). For detecting an 85 Hz frequency, the signal must be at least 0.012 seconds long. In order to acquire enough data to perform the windowing process for the windowed envelope without a leakage effect, a signal length of 0.11 seconds might not be the best solution. Since the accuracies of 0.2 seconds and 0.7 seconds are nearly equal, the shorter signal length of 0.2 seconds is used to get a more universal solution for the intermediate domain.

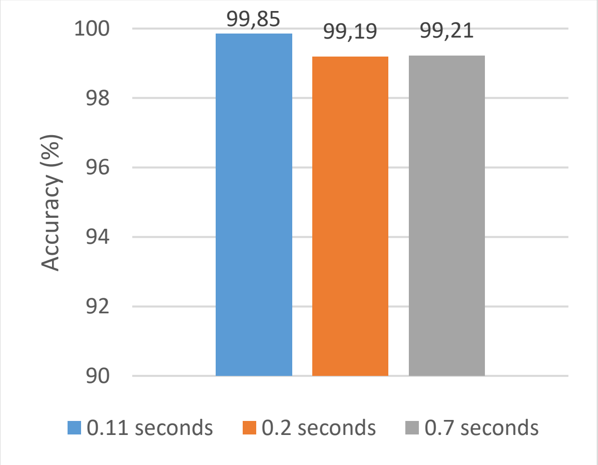

Figure 26: Accuracies of samples of different lengths. All samples are generated with the windowed envelope method. All accuracies are between 99% and 100%. The test setup is described in Appendix A.2.2.

4.4.4 Conclusion

This chapter showed that it is possible to combine classical frequency and time-frequency domain transformations to be used for the feature extraction of sensor data. Therefore, the existing techniques of envelope analysis and SFTF have been combined into a new windowed envelope analysis. With the help of different test scenarios, it has been shown that for the given use case of the fault classification of bearings of the CWRU dataset, this combination delivers better accuracies than other state-of-the-art techniques such as S-transform or HHT (see Figure 25). The result of the windowed envelope for a 0.2 second signal of a bearing is illustrated in Figure 27.

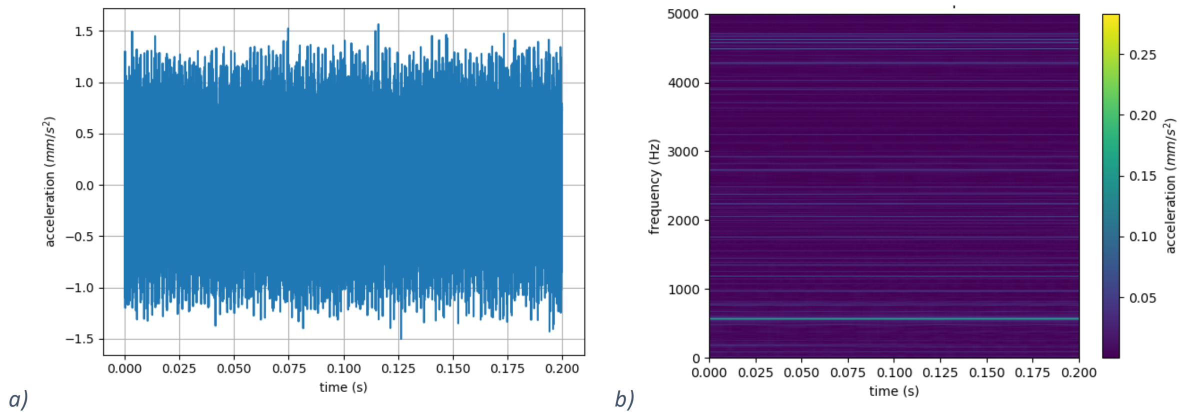

Figure 27: The raw sensor signal as input (a) and the output (b) of a windowed envelope transform. (b) shows that numerous frequencies appear in the converted signal. For instance, the transformed signal shows that there are frequencies with a high amplitude at 800 Hz as well as several frequencies around 4500 Hz. These frequencies are not only related to the fault, but also belong to frequencies from other components of the process or to the natural frequencies of the bearing itself (cf. [133]).

4.5 De-noising

4.5.1 Introduction

As a next step for creating the proposed intermediate domain, the frequency domain data must be de-noised. This step occurs after the signal processing because the frequency data can be more easily de-noised once it is transformed into this domain (e.g., band-pass filtering) [165]. The de-noising is important for sensor data in mechanical systems. The sensor data is a mixture of background noise and the fault signature in practice. Because of the background noise, it is more difficult to identify the fault signature [166]. Current research proposes different methods for the time, frequency, and time-frequency domains. An efficient method for de-noising in the time domain is to use averaging methods for periodic signals [167]. For data in the frequency domain, filtering methods such as band-pass filtering and low-pass filtering are typical ways to eliminate noise outside the manually defined frequencies of interest [165]. There are also methods for the time-frequency domain, such as singular value decomposition [105] or wavelet transform-based filters [168]. All these filtering methods need additional parameters for optimizing the results of the de-noising process. These parameters are often based on the empirical experiences of specialists [165].

4.5.2 Bandpass

For the automation of the filtering process, domain knowledge can be used to automatically select the relevant frequencies. Therefore, the frequency bands around the characteristic fault frequencies as well as their harmonics can be used. These frequencies can be automatically calculated by knowing the exact mechanical layout and the current rotational speed of the bearing (see Section 3.2.4). Even though the characteristic frequencies can be calculated exactly, frequency bands are used for the proposed intermedia domain to account for frequency deviations. These frequency deviations can be caused by wear-out of the bearing parts but also by manufacturing tolerances and uncertainties regarding the actual rotational speed [130]. It is essential that the used frequency band has the correct width. A too-narrow band will remove the above-described tolerance frequencies, and a too-wide band can include unwanted frequencies. This is especially the case for high rotational speeds since dimensional differences of the bearing parts have a more significant influence here. This is because the fault frequency is calculated by multiplying the speed with a fault factor that is specific for the component dimensions (e.g., Eq. (15)- (18)). Therefore, the nominal frequency can be different from the actual one.

A real-world dataset of grinding spindles with noise is used to select an appropriate bandpass width (see Appendix A.2.3). As shown in Figure 28, a bandpass width of 10 Hz provides the best accuracy. It is therefore selected in the presented approach.

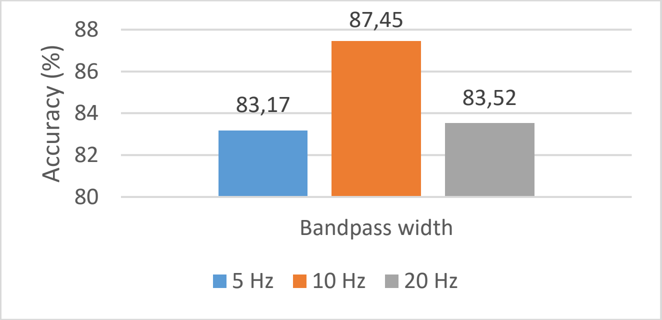

Figure 28: Accuracies for different bandpass widths of a frequency-selective filter, which uses four harmonics. The best accuracy is achieved with a width of 10 Hz. The lower accuracy for 5 Hz might be related to the fact that the actual fault frequency in some samples defers more than 5 Hz through wear-out and manufacturing tolerances. On the other hand, the 20 Hz frequency band might be too wide, so that noise gets into the band, which can actually be filtered out.

4.5.3 Harmonics

As described in Section 2.3.3.2, the harmonics of the frequency are also important because when an envelope analysis is done, the harmonics also have an appreciable amplitude. In some cases, the second and third harmonics can even have a higher amplitude than the original frequency [169]. Therefore, the proposed intermediate domain stacks the first *n* harmonics of each fault frequency band in layers on top of each other (see Figure 30 b).

Using the characteristic frequencies and their harmonics may lead to another advantage of this approach: It can consider different rotational speeds. For the use case of bearings, all error frequencies depend on the rotational speed (see Selection 3.2.4). Using the method described above, the frequency band is always located at the same image position. This makes transfer learning easier since the layout of the images is always identical.

To validate the benefits of the use of harmonics, especially for bearings under different process conditions, the same bearing dataset as for the estimation of the bandpass width is used (see Appendix A.2.4). As can be seen in Figure 29, the accuracy of the windowed envelope drops from more than 99% for bearings used in a test stand (which has been shown above in Figure 26) to 75% using a dataset of a spindle, which is used at higher rotational speeds and also in a noisier environment. It also shows that the accuracy can be increased to 87% using a frequency-selective filter with four harmonics. This indicates that the frequency-selective filter, which uses the first four harmonics, is a good choice to increase the accuracy without manually fine-tuning the parameters. The results also indicate that the information outside the selected frequency bands is irrelevant for the fault classification and can therefore be removed.

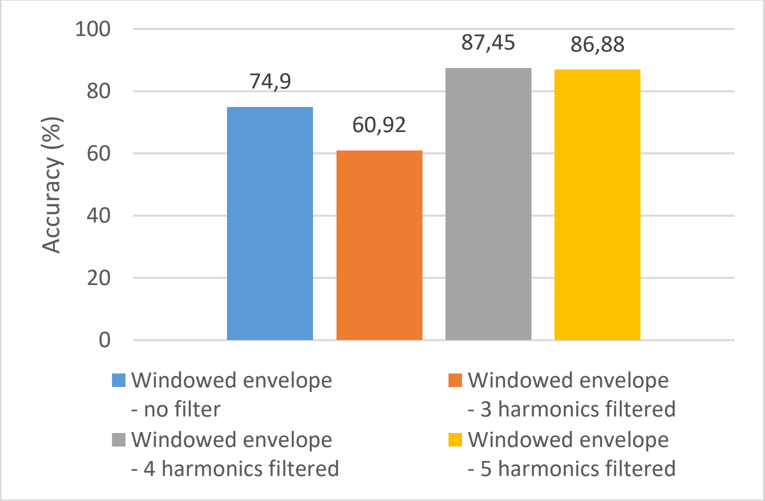

Figure 29: Accuracies for bearing fault classification with the help of a windowed envelope in a noisy environment. The highest accuracy can be reached by applying a frequency-selective filter with four harmonics (see Appendix A.2.4).

4.5.4 Conclusion

This chapter showed the effectiveness of de-noising based on a frequency-selective filter. This filter takes care of the fault frequencies and their harmonics. Therefore, as presented in Figure 30, the input, which is, in this case, a windowed envelope, is split into layers for each fault frequency. Only these layers remain in the output.

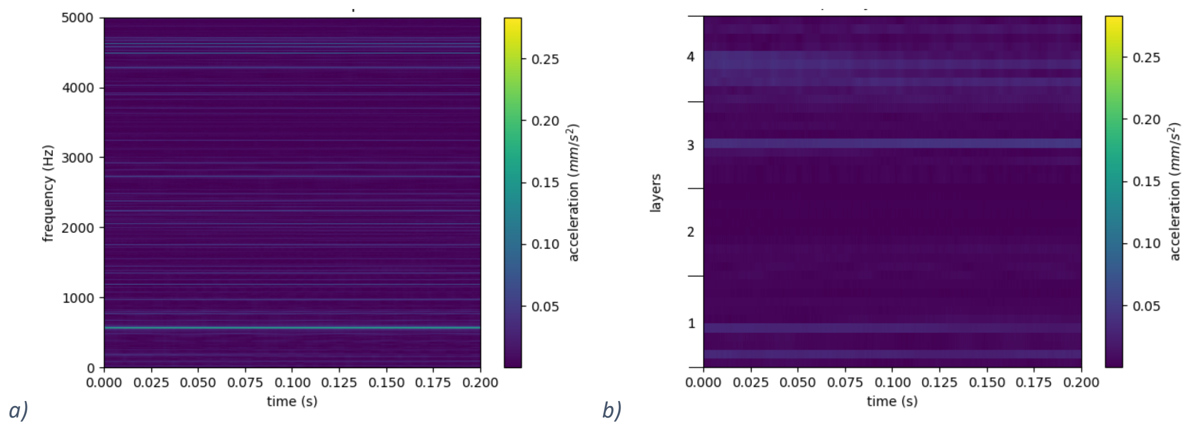

Figure 30: These figures show the windowed envelope as input (a) and the output (b) of the frequency-selective filter. In (b) only the relevant layers remain. There is a layer for each fault frequency of the bearing. Each layer contains a frequency band around its fault frequency. This is done for n harmonics. Additional frequencies that are not related to the fault are filtered out. In this case, there are four layers for four different fault frequencies including four harmonics for each. It is clearly visible that the high amplitude, which occurs at approx. 800 Hz in (a) is filtered out and no longer appears in (b) (cf. [133]).

4.6 Preparation

The last step of the creation of the intermediate domain is to prepare the image to be used for the classification task. Therefore, the images first have to be normalized. This step is advised when features have different ranges. For example, this can happen when performing transfer learning from one type to another type. However, the normalization of the data in a range between 0.0 and 1.0 is also common practice to increase the training speed, especially when using gradient descent algorithms [6]. These algorithms are one of the most commonly used optimization algorithms in the machine learning field, especially for CNNs [170]. In addition, some loss functions are only valid in this range. For instance, the commonly used cross-entropy is defined by a negative log and therefore is only valid in this range [6]. Afterward, the image must be resized to fit the input size of the machine learning model. This is especially important if the target and the source domain data were recorded with different sampling rates, which leads to different output sizes. If a sensor has a higher resolution, more measurement values are recorded in a defined time sequence, leading to a higher resolution for the transformed image and, therefore, a larger image size in terms of pixels.

There are several possible sizes for the image to be rescaled. One recommendation is that an input size of a CNN should be divisible many times by two, such as 32, 64, 128 [171]. If the classification accuracy between two resolutions is equal, the smaller one should be used, since smaller images are beneficial in that they require less memory. Smaller images also have benefits during the training phase of a CNN because smaller images lead to smaller layers, which in turn leads to less trainable parameters. This increases the training speed.

In order to determine a suitable resolution, a comparison of the image sizes of 64 pixels and 128 pixels with the two previously used datasets has been made (see Appendix A.2.5). The results, which are shown in Figure 31, have shown that, on average, the 64x64 pixels images perform better.

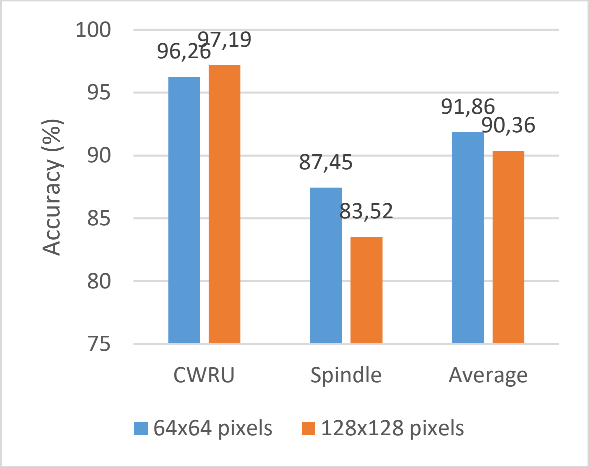

Figure 31: Accuracies of different sizes of the input image. For the CWRU dataset, a 128x128 pixel image has a slightly better accuracy. For the Spindle dataset, the 64x64 pixel image performs better. On average, the 64x64 pixel image has the better accuracy.

The same conclusion was also drawn in the external research of Verstraete et al. [141]. They showed that increasing the pixel density does not lead to any significant change in accuracy for their use case. The researchers compared the accuracies of STFT spectrograms generated by bearing defects of the CWRU dataset of 32x32 pixel images with the size of 96x96 pixel images. Here the difference in accuracy was only 1.5% (32x32 pixels: 98.0% and 96x96 pixels: 99.5%). Therefore, in this thesis, the lower image size of 64x64 pixels is used as input for a CNN. This leads to a rescaled image of 64 pixels for the timeline in the x-direction and 64 pixels for all frequency layers together in the y-direction.

To summarize, the preparation is based on two steps. The first step is to normalize the images, and the second is to rescale the images. The resulting image (as shown in Figure 32b) is the final input for a machine learning algorithm, such as a CNN.

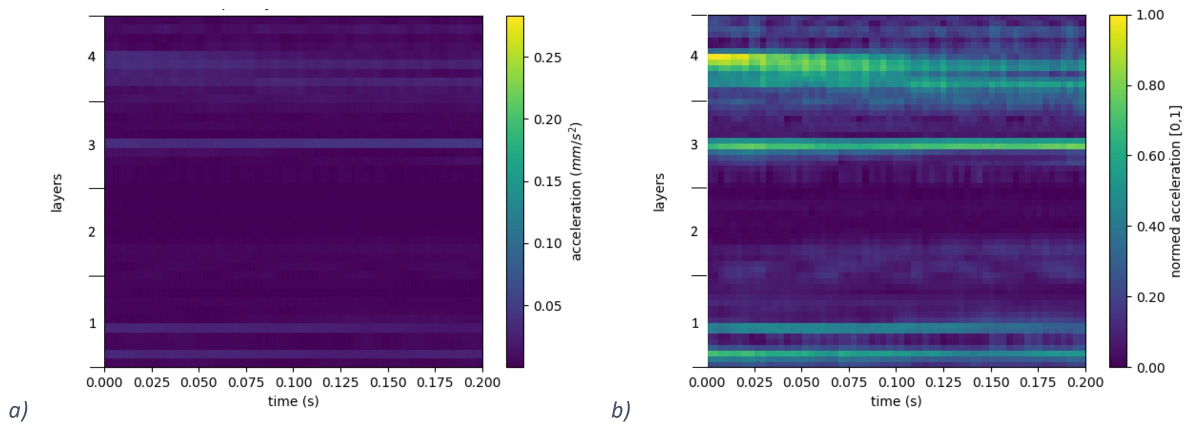

Figure 32: The image of the frequency-selective filter as input (a) of the preparation process and its output (b), which is the final intermediate domain image. The preparation process performs normalizing and rescaling on the input. The output is now normalized in a range between 0.0 and 1.0. In addition, it is recognizable that the resolution has been reduced to 64x64 pixels because the image now appears pixelated (cf. [133]).

4.7 Supplements for the Use Case of Bearing Fault Classification

The derivation of the intermediate domain was verified with the help of test scenarios for the bearing fault classification. The parameters, which are needed to create the intermediate domain, were already selected in the previous sections. As shown in Table 9, these are the use of signal segments of 0.2 seconds to use the filtered windowed envelope with four harmonics and a bandpass width of 10 Hz, and the use of a 64x64 pixel image. This chapter provides additional information for the use case of bearing faults. As stated in Section 3.2.4, the process parameter with the most influence on the error pattern of bearings is the rotational speed. Therefore, any changes in the rotational speed, which may be needed for the different requirements of the process, may lead to a different fault frequency. However, it can be assumed that the speed can be kept constant during the analysis of the bearing. This can be achieved by diagnostic routines, which analyze the bearing state at defined rotational speeds (see Section 4.3). This is also proposed by commercial applications [130].

The de-noising of the data with the help of the frequency-selective filter is especially important for the use case of a bearing inside of a spindle because there is significant noise. As shown in Figure 34, the result of the intermediate domain is an image with 16 areas: four harmonics for each of the four characteristic frequencies (cage, ball, and inner and outer ring). The external frequencies outside of these frequency bands are removed. Figure 33 illustrates a bearing with an outer ring defect, including these external frequencies of other components, which appear when a spindle is used inside a machine. These external frequencies are mainly noise from unknown origin. However, in the case of spindle data, the internal research of Erwin Junker Maschinenfabrik GmbH has shown that there are two well-known external frequencies. The first is the frequency of the variable-frequency drive, which is used to model an output frequency to operate the three-phase drives of the spindle at variable speed. Typically, this frequency is next to 4 kHz. The second frequency is based on the control cycle of the current control used to guarantee a constant current for the input of the variable-frequency drive. In this example, the current control clock is set to 125 µs, which results in a frequency of 8 kHz. In addition, there are overlays of the stator frequency and its harmonics. This frequency is calculated as a function of the rotational speed of the spindle. A typical frequency range is approximately 100 Hz.

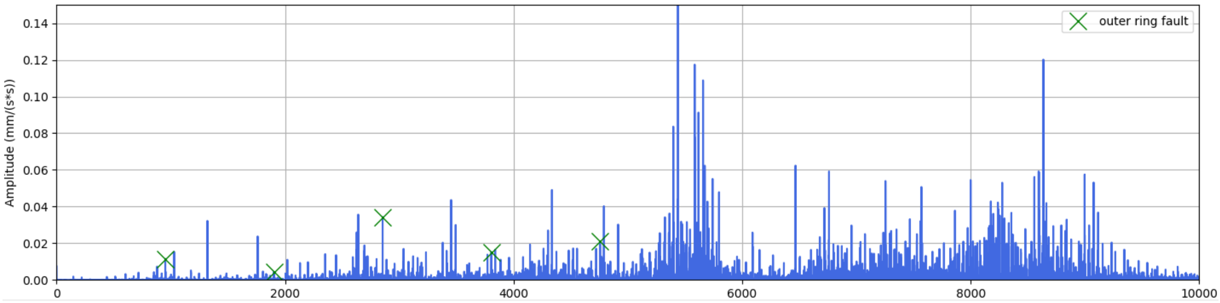

Figure 33: Frequency domain of a bearing with an outer ring defect and "natural" noise of a spindle in use. The characteristic frequency of the outer ring defect plus its harmonics are marked with a green crosshair.

In Section 4.5.3, the suggestion to use four harmonics based on the achieved accuracy was made. There is also a significant practical disadvantage in using the fifth or higher harmonics: As introduced in Section 3.2.5, current accelerometers for industrial use can record data with a maximal resolution between 10,000 Hz and 50,000 Hz, but most can only record up to 10,000 Hz. The problem is that a standard spindle for grinding machines has high fault frequencies when the spindle works at its maximum speed. Realistic values for such a spindle of the Junker Maschinenfabrik GmbH are a maximum rotational speed of 15,000 rpm and a bearing inside the spindle with the following parameters: 25 balls, 4 mm ball diameter, 45 mm pitch diameter, and a contact angle of 15°. By inserting these parameters into Eq. (17), a fault frequency for the inner ring fault of 3,393 Hz can be calculated. For an intermediate domain that uses four harmonics, the maximum fault frequency is four times 3,393 Hz, which is 13,572 Hz. If sensors with low resolutions are used, capturing the fault signal at a maximum rotational speed is impossible. If a higher number of harmonics were used, the speed range in which faults can be evaluated would be decreased further. Independent of the number of the used harmonics, it is therefore recommended, as mentioned above and in Section 4.3, to use specific predictive maintenance routines. These routines should generate a fixed fault frequency based on a constant rotational speed but also generate fault frequencies within the operating range of the sensors used.

Table 9: The different parameters of the intermediate domain.

Parameter	Value
Signal length	0.2 seconds
Bandpass with	10 Hz
Used harmonics	4
Output size	64x64
Layers	Cage fault, inner ring fault, outer ring fault, ball fault

The intermediate domain image for bearings consists of 16 frequency layers for four fault types. The resulting final intermediate domain image for a bearing is shown in Figure 34. This image is based on the settings shown in Table 9.

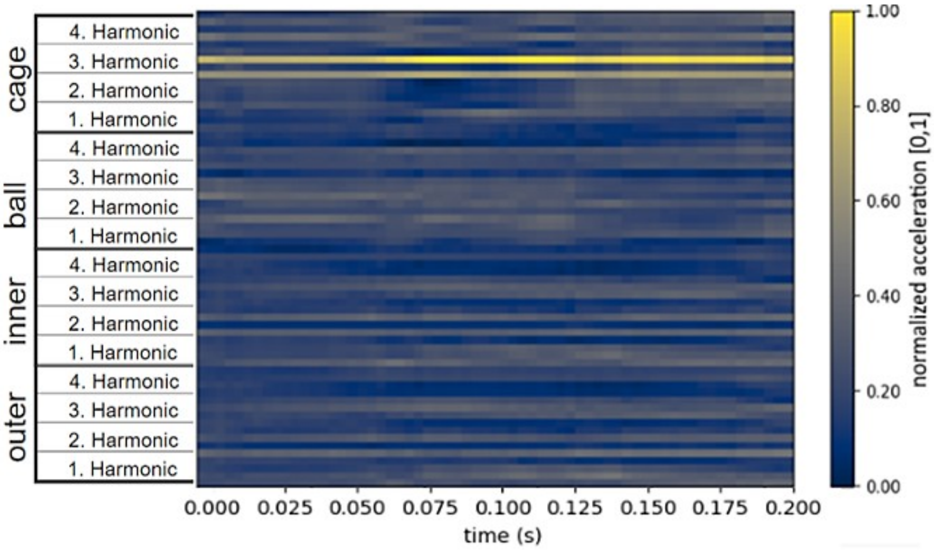

Figure 34: Intermediate domain image for four different fault types. For each fault type, the first four harmonics are used and stacked on top of each other [133].

4.8 Generalization

The proposed intermediate domain was verified for sensor signals used for the bearing fault detection with the help of their fault frequencies. However, bearings are not the only components that can be analyzed with the help of the fault frequencies and their harmonics. If a different component than a bearing is used, the intermediate domain parameters might need to be modified. These are signal processing parameters: the number of layers, which have to be chosen according to the available fault frequencies, the number of used harmonics, and the bandwidth. Another parameter is the signal input parameter in the form of the signal length used for one window envelope. Finally, the output size must also be re-evaluated.

A different component, which may be used with the intermediate domain, is, for instance, a gear. Gear fault classifications are also made based on fault frequencies. Here, the characteristic frequencies are the gear rotational frequency, the pinion rotational frequency, and the gear mesh frequency [172]. Some modifications must be made in the case of gears. At the very least, the number of layers for the intermediate domain must be changed since gears only have three different fault frequencies, and bearings have four different fault frequencies. This leads to having only three layers in the intermediate domain. In addition, it may be necessary for the bandwidth of the frequency-selective filter to be changed. This assumption is based on the idea that gears may have different fault frequencies and manufacturing tolerances. The hypothesis of the application to gears is based on theoretical considerations and has not been verified based on a reference data set.

4.9 Conclusion

This chapter has presented a new feature extraction method based on an intermediate domain for bearings. The presented intermediate domain creates an abstraction of the raw sensor data by using a new time-frequency transform that is called windowed envelope. This windowed envelope is filtered by frequency bands of the harmonics of the characteristic fault frequencies. All parameters of the intermediate domain are empirically determined in different test scenarios. One possible scenario to use the intermediate domain is to use it to analyze bearings in periodical test cycles. During this cycle, it can be used for classification and RUL tasks.

The usability of the intermediate domain will be verified later in the thesis in the form of different case studies. On the one hand, this concerns the stability of the intermediate domain for different bearing types, which is verified by the case studies in Section 7.2 and Section 7.3 for classification tasks and in Section 7.4 for an RUL task. On the other hand, the ability of the intermediate domain to improve transfer learning between different datasets is analyzed in detail in Section 7.2. Therefore, the intermediate domain is benchmarked against an HHT and an S-transform. In this case, the intermediate domain can increase the classification accuracy by more than 30%. In addition, with the help of a data-driven approach such as a CNN, the remaining data can be analyzed without manually fine-tuning [13]. For those reasons, the chosen intermedia domain-based approach is also the answer to RQ3, which asks for a stable feature extraction method for bearings that is well suited for transfer learning. Furthermore, it is worth mentioning that the intermediate domain should also be suitable for other components with periodic motions, as asked in the general problem definition in Section 4.1 and discussed in Section 4.8.

In addition to answering RQ3, the presented intermediate domain also provides an answer to RC3. This RC, which demands an answer on how existing methods can be combined and optimized, is answered by the intermediate domain itself. Here, particular focus should be paid to the new time-frequency transform, which is called windowed envelope. This transform combines the existing envelope spectral analysis and the STFT.

5 Transfer Learning Approach for Classification

5.1 Introduction

The current state of the art for predictive maintenance solutions in general and for the classification of bearing defects in particular was shown in Chapters 2 and 3. As also mentioned in Chapter 1, solutions are needed for real-life predictive maintenance scenarios, where datasets are often small and unlabeled or partly labeled for new machines or new types of a component [9]. In addition, these small target datasets may also be based on different process conditions. However, labeled datasets of a different component type or a different machine type may be present in large numbers. Therefore, a research gap for a solution that fits the following requirements exists:

- A general solution that can be applied to different components of a machine. These components must meet the criterion that they are moving parts that make periodic movements. In addition, it must be possible to assign their condition to specific condition groups based on the measurement values of sensors.
- It shall have superior classification accuracy compared to the current state-of-the-art techniques, especially for real-world scenarios encountering noise.
- An important requirement is the transferability of the results. There are two relevant types of transfer. The first is when the classification process uses datasets from a different type of the same component to enhance the classification accuracy. The second is when, within the same dataset, samples of different process parameters exist. The solution shall handle both transfer types at the same time.

The above problem definition is very general and is suitable for various components. However, as also required in RQ1, the specific use case of bearings will be addressed in the following sections. RQ1 asks for a new classification method that can take benefits of a dataset of a different bearing type for a partly labeled target dataset that is collected under different process conditions. This is a common use case since bearings are a component of many machines; for instance, they are used inside spindles. As mentioned in Section 3.2.3, there are many different spindles, which are all specific for a particular use case. In addition, bearings are often used under different process parameters that are related to the produced workpiece. A solution that fulfills the aforementioned requirements is particularly important because of the difficulty of getting many samples for training the different machine learning approaches for new bearing types or spindle setups.

As stated in Section 2.4, a possible solution for this RQ might be to use a transfer learning-based approach, that uses the intermediate domain images of Chapter 4 as input for the classification. This feature extraction technique could also bring advantages for transfer learning tasks (see Section 4.3).

CNNs have been shown to be a suitable technique for the classification of images (see Section 2.2.4). Additionally, for the semi-supervised transfer learning case, the unique properties of the intermediate domain might be used to apply a hybrid transfer algorithm (using contextual knowledge).

This chapter proposes a novel solution to verify this hypothesis in a complete process chain (see Figure 35). The general layout of this approach corresponds to the standard procedure for classifications based on sensor values, which start with the raw sensor data as input. This data is then preprocessed by a time-frequency transform and afterward used with a CNN for the final classification task (see Section 3.3.2).

The proposed solution implements these steps as follows: as a first step, the preprocessing is performed by the intermediate domain introduced in Chapter 4. This intermediate domain is a hybrid approach, which also considers domain knowledge to get a better feature representation, especially in the case of transfer learning. This is achieved, amongst other techniques, by a modification of the current model-based state-of-the-art technique for bearing analysis (envelope analysis), which extracts features in the input data in only relevant frequency areas. Afterward, this intermediate domain data is used as input for a proposed transfer learning architecture based on the current state-of-the-art technique for classification: CNN (Section 5.4). For the training of the CNN, two different transfer learning approaches are presented: One approach processes labeled and unlabeled data in parallel. The other approach processes the data sequentially, whereby first unlabeled data is used, followed by labeled data. As the last step, a new domain adaptation loss function, called the LMMD, is presented. The LMMD is designed to take care of the characteristics of the new intermediate domain (Section 5.5). In summary, this solution is a new hybrid solution that combines the advantages of model-based and machine-learning-based techniques. For each part of the solution, first, the reason for its design decisions is introduced, followed by a detailed explanation.

The proposed solution has the benefit that it can be used out of the box. Due to its ability to transfer knowledge to a different bearing type under different conditions, it must also fit the other bearing type or conditions, otherwise, the accuracy would be low after transfer learning. In contrast to this are new solutions without the ability to use transfer learning. Therefore, there is always the need for experts [165]. This need is valid for feature extraction as well as for machine learning itself. Feature extraction methods, like the one presented in Section 2.3, have parameters such as, for instance, the number of scales for wavelets (see Section 2.3.4.3), which need to be chosen correctly to obtain correct results. The same goes for the machine learning algorithm. There are also parameters that must be optimized for the given use case, like the different layers of a CNN (see Section 2.2.4).

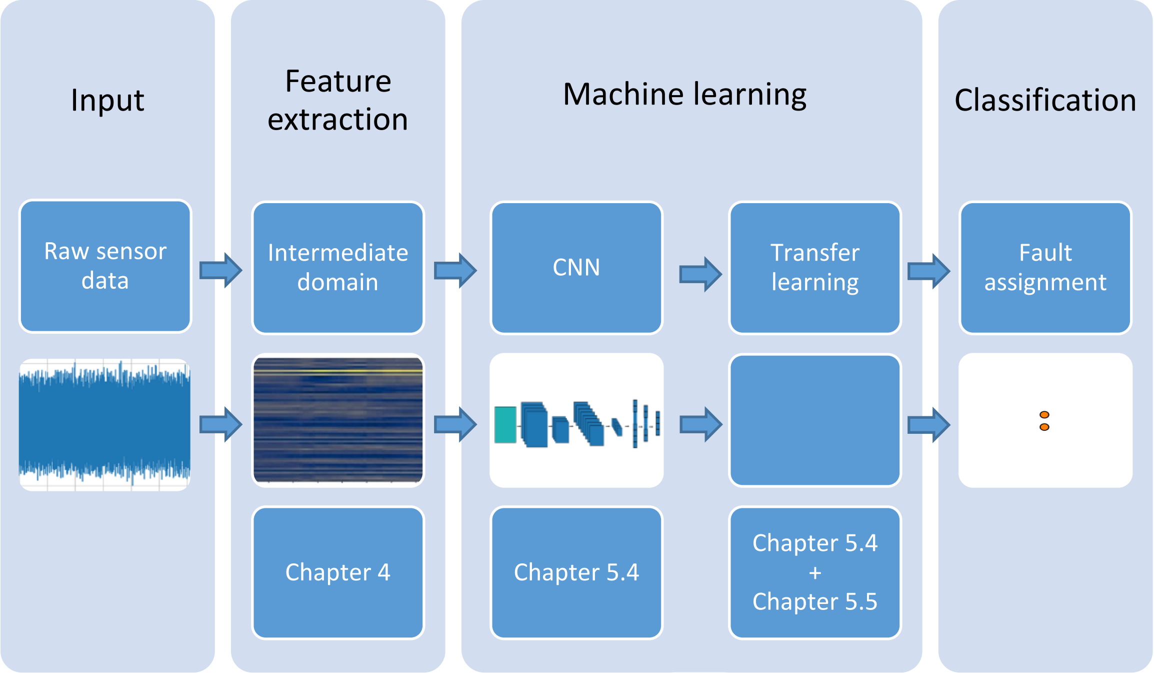

Figure 35: The different steps for a transfer learning-based classification approach. The input, which is the raw sensor data, is preprocessed with a feature extraction technique such as an intermediate domain. Afterward, the extracted features are used for the machine learning process based on CNN with transfer learning. The trained model is finally used for classification tasks.

The presented approach is verified in detail with three bearing-related case studies in Section 7.2 and a benchmark in Section 7.3. This approach and the corresponding case studies have also been summarized and published in a research paper [133].

5.2 Validation Context

All design decisions of this chapter are based on empirical test scenarios based on bearing datasets. Therefore, each scenario is validated with the classification accuracy based on the Case Western Reserve University and the Junker Maschinenfabrik GmbH datasets and their mean accuracy (see Section 3.3.2.2). The other test parameters are identical to those presented in Chapter 4. One of them is the split ratio of the datasets. For each test scenario, the dataset is split based on a ratio of 70% training data and 30% test data. The result of one test scenario is the mean of two runs with a different training and test data split. This split is identical between all test cases in one test scenario and is given in Appendix A.1.1 in Table 29. The parameters of the training itself are listed in Table 10.

Table 10: Parameters used during the training of the neural network.

Parameter	Value	Reason
Optimizer	Adam	Empirical tests and recommendations such as [6]
Learning rate	0.00003	Empirical tests.
Batch size	150	Empirical tests.
Iterations	200	Empirical tests.
Loss function	Cross-entropy	See Section 5.4.2.4

5.3 Classification Support by Transfer Learning

There are many approaches for the different classification tasks in the context of predictive maintenance, such as fault classification of bearings [54], gears [172], and motors [173]. As mentioned in Section 2.2.2, these approaches can be divided into approaches with and without machine learning. Machine learning-based approaches have the benefit of outperforming non-machine learning based approaches if the classification task is complex (see Section 2.2.1). In this case, by using non-machine learning-based methods, there is often the need for a high degree of fine-tuning, even for only small process changes. Because of the often-complex dependencies on various conditions and process parameters, this fine-tuning can only be carried out by process experts. There is even a possibility that no solution can be found at all. Therefore, it is recommended to use machine learning techniques where it is often sufficient to retrain a predefined ML model for a similar task [1]. However, as explained in Section 2.4.1, the major drawback of machine learning techniques in general and deep learning methods in particular, is that they require a large dataset for training, which is not available in many scenarios [9]. This problem can be solved through transfer learning. As mentioned in Section 2.4.3.5, there are different approaches to transfer learning. For the given problem of a small, partly labeled, or even unlabeled dataset, domain adaptation, a subfield of transductive transfer learning, is particularly important. Domain adaption is a transfer learning approach for using a large, labeled dataset in the source domain and a dataset of a different domain with the same learning tasks as the target domain. It can be said that for the given predictive maintenance use case, domain adaption is a suitable method because all the described preconditions are true: it involves the detection of errors, often with complex dependencies, and the use case of small unlabeled or partially labeled data in the target domain.

5.4 Proposed Transfer Learning Architecture

5.4.1 Introduction

After the feature extraction of the raw sensor to 2D images using the intermediate domain of Chapter 4 , the next step is to use them as input for a machine learning algorithm. As stated in Section 2.2.2, the most popular and successful approaches for classification are based on CNNs and SVMs. Different surveys compare the relevant machine learning approaches in the context of bearing fault classification [54, 131]. These approaches are based on different techniques such as SVM and CNN and are used for supervised learning in one domain as well as for transfer learning between different domains. The surveys point out that all algorithms could deliver similar results in supervised learning. For the use case of transfer learning, nearly all approaches are based on CNNs. Only a few are based on other techniques, such as an SVM, as is done by Li et al. [174], who combined an SVM with the

MMD for transfer learning, or autoencoders, as is done by Zhiyi et al. [147]. For CNNs, as stated in Section 2.2.4, the first layers are for the feature extraction of the CNN input data and the last layers are used for classification. In the case of transfer learning, the existing CNN can be fine-tuned as described in Section 2.4.4. Thereby, large parts of the feature extraction layers of the CNN can be taken over from the model trained with source domain data. This makes CNNs well suited for transfer learning, which also leads to the high number of CNN approaches for transfer learning. Other techniques used for transfer learning do not have the benefit of distinguishing between feature extraction and classification. The good performance of CNNs for transfer learning tasks, especially when using sensor data, is not limited to images as input. As introduced theoretically in Section 2.2.4 and outlined by practical examples in Sections 3.3.2.4 3 and 3.3.2.5, there are also solutions that use the raw sensor data as 1D input for a CNN.

Based on this derivation, the proposed solution is based on a CNN. The usage of the CNN is also proposed because it makes it possible to use the images of the proposed intermediate domain as input, making it a hybrid approach. Therefore, the physical parameters of the bearings are also taken into account. This results in a solution that uses two feature extraction layers: the intermediate domain itself and the lower convolutional layers of the CNN.

The proposed CNN model is an approach that should be usable for supervised, semi-supervised, and unsupervised learning. In addition, the model is usable for two different kinds of semi-supervised learning. In Section 5.4.3, the first transfer learning approach (TLA) that performs transfer learning with labeled and unlabeled data of the target domain simultaneously (later referenced as Transfer Learning Approach 1 (TLA1)) is presented. This is followed by Section 5.4.4, which presents the second approach, which first uses the unlabeled data of the target domain for training, followed by a training step with only labeled data of the target domain (later referenced as Transfer Learning Approach 2 (TLA2)).

5.4.2 CNN Model

5.4.2.1 Introduction

Every CNN model is defined by several parameters. This chapter derives the important ones and the design decisions of the proposed CNN model. Therefore, first, the used layers of the model are introduced (Section 5.4.2.2). This is followed by the CNN hyperparameters window size and dropout factor (Section 5.4.2.3). Finally, the used loss function is evaluated (Section 5.4.2.4).

5.4.2.2 CNN Layers

The layout of the used CNN is based on the architecture of a CNN introduced by Verstraete et al. [141]. The authors presented a CNN that is superior to other CNNs for fault classification based on sensor data. As described in Section 3.3.2.3, they used the CNN to classify bearing faults on different 2D images of time-frequency transforms (HHT, Wavelet, and STF). The images are processed by a CNN, which first extracts the features and finally assigns the image to one of the four health states of bearings. The architecture of this CNN is based on double convolutional layers. As displayed in Figure 36, two convolutional layers are used in series, in contrast to the traditional layout, where a convolutional layer is followed by a pooling layer. According to the authors, this layout increases the significance of the features through an additional nonlinearity. Each convolutional layer increases the nonlinearity through its nonlinear activation function. A nonlinearity decision function is essential for making complex nonlinear decisions [142]. These decisions can be complex predictive maintenance tasks. Their CNN architecture uses these double convolutional layers three times with a different number of filters on each layer (32, 64, and 128). They are followed by three fully connected layers, where the last layer is used for the classification of different fault types.

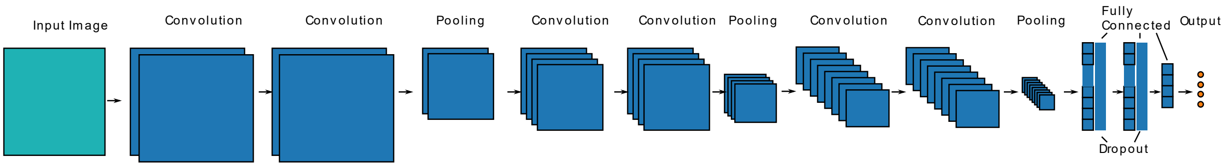

Figure 36: CNN architecture proposed by Verstraete et al. [141]: three consecutive blocks of a combination of two convolutional layers and pooling layers are used. In each block, a different number of filters is used. Afterward, two fully connected layers with a dropout layer after each feed the last fully connected layer, which is used for the classification.

The decision to use the double convolutional layer based architecture of Verstraete et al. [141] has also been verified with two test cases with different bearing datasets in Appendix A.3.1. The results, shown in Figure 37, pointed out that accuracy is improved in both cases with the help of this architecture.

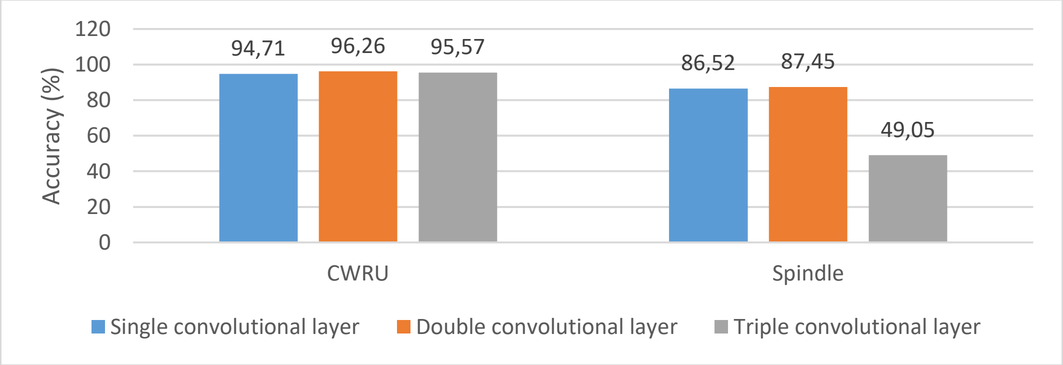

Figure 37: Comparison of a different number of convolutional layers in series. Two different test cases have been used. For both tested datasets, the double layer approach has the best accuracy.

5.4.2.3 CNN Hyperparameters

In addition to the layout itself, other hyperparameters, such as window size, activation function, and dropout ratio, are also important. The proposed model uses a window size of 3x3 for each convolutional layer, combined with the activation function rectified linear unit (ReLU). A window size of 3x3 has been selected because a window size of 1x1 only makes sense for dimensionality reduction, since a 1x1 window cannot use any information from the neighboring pixels. In addition, a window size of 2x2 and 4x4 should not be used because those filters do not use the pixels of the input layer symmetrically, which leads to distortions across the layers. Window sizes larger than 4x4 are not recommended because they lead to long training times [175]. The ReLu, which is defined as $ReLU(x) = \max(0, x)$, has been chosen because it reduces the likelihood of the gradient vanishing (see Section 2.2.5) and results in faster learning through its constant gradient [6]. A dropout layer with a dropout ratio of 50% was inserted between each of the fully connected layers to prevent overfitting and minimize training error. The factor of 50% is used based on literature suggestions such as [6]. In addition, a test case has been used to evaluate the influence of the different dropout factors and the suggested value of 50% (see Appendix A.3.2). The results, which are shown in Figure 38, reveal that there are hardly any accuracy changes when using different dropout values. Nevertheless, to be prepared for possible overfitting with other datasets, a dropout factor of 50% was chosen according to the literature's recommendations. This means that during each training step, 50% of the neurons are completely ignored [6].

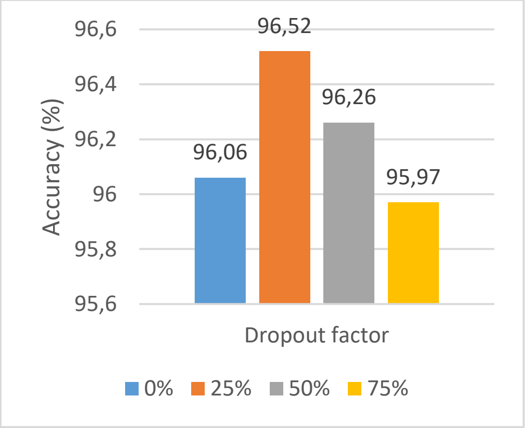

Figure 38: Different values for the dropout factor in the fully connected layers of the CNN. There is no significant difference between the accuracies.

5.4.2.4 Loss Function

During the training phase, a loss function must be chosen. As stated in Section 4.6, cross-entropy is a commonly used loss function for multiclass learning of CNNs. This loss function should only be changed when problems arise using this algorithm [176]. Another efficient loss function is the Kullback Leibler divergence, which measures the differences between two probability distributions. It is often used when the training goal is to recover an input signal, as is the case with autoencoders [6]. To validate that the loss function, which is proposed by literature, is also the best for this use case, both loss functions have been compared (see Appendix A.3.3). The result, which is also displayed in Figure 39, indicates that neither is superior. One time cross-entropy performs a little better, and in the other case, the Kullback Leibler divergence emerges. Therefore, according to the literature, cross-entropy has been selected as the loss function for the proposed CNN.

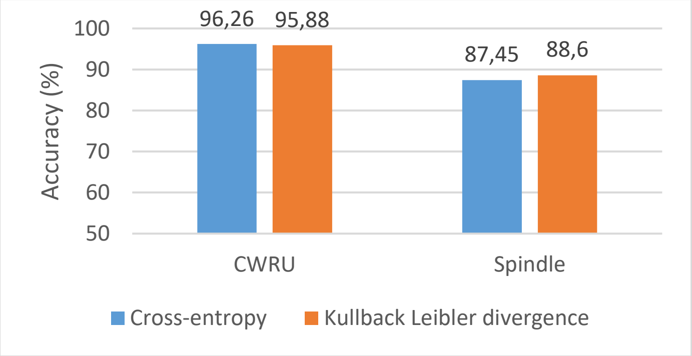

Figure 39: Resulting accuracies of CNN trainings with cross-entropy and Kullback Leiber divergence. Neither is superior.

5.4.2.5 Conclusion

The previous sections have introduced the used CNN model and its hyperparameters. A detailed summary of this CNN model for the 64x64 pixel images of the intermediate domain is shown in Table 11. This model has a total amount of 1,116,236 trainable parameters. This architecture can be directly used for supervised training.

Table 11: This table shows the detailed architecture of the proposed CNN with the layer types and their parameters. Each layer of the CNN is listed with its type, the output shape of each layer (none has to be replaced with the number of images that are used in a batch and is therefore dependent on the training parameter batch size), the number of trainable parameters, and used activation functions.

Layer	Type	Output Shape	Trainable Parameters	Activation
0	InputLayer	(None, 64, 64, 1)	0	
1	Conv2D	(None, 64, 64, 32)	320	ReLu
2	Conv2D	(None, 64, 64, 32)	9248	ReLu
3	MaxPooling2D	(None, 32, 32, 32)	0	
4	Conv2D	(None, 32, 32, 64)	18496	ReLu
5	Conv2D	(None, 32, 32, 64)	36928	ReLu
6	MaxPooling2D	(None, 16, 16, 64)	0	
7	Conv2D	(None, 16, 16, 128)	73856	ReLu
8	Conv2D	(None, 16, 16, 128)	147584	ReLu
9	MaxPooling2D	(None, 8, 8, 128)	0	
10	Flatten	(None, 8192)	0	
11	Dense (FC1)	(None, 100)	819300	ReLu
12	Dropout	(None, 100)	0	
13	Dense (FC2)	(None, 100)	10100	ReLu
14	Dropout	(None, 100)	0	
15	Dense (FC3)	(None, 4)	404	Softmax

5.4.3 Transfer Learning Approach

When performing transfer learning with unlabeled or only partly labeled data in a different domain with the same task, one speaks of domain adaption (see Section 2.4.2). As described in Section 2.4.4

regarding the use case of CNNs, this can be done with discrepancy-based approaches. These approaches all follow the same procedure by duplicating the used CNN, as shown in Figure 40. In doing so, two paths are created. One uses images of the source dataset while the other uses images of the target dataset. The CNN must first be trained during a supervised training phase with only labeled source domain data before doing the transfer learning. A fixed feature extraction can be done for the transfer learning itself because the source and the target domain images are expected to be similar due to the intermediate domain. Therefore, only the classification part must be adopted (see Section 2.4.4). Consequently, the weights and biases of the convolutional layers are frozen and shared between the source and the target path. The output of the fully connected layers of each source and target domain pair is used for calculating an addition loss value that is called L_{FC1} for layer 1 and L_{FC2} for layer 2. The loss function used for this purpose must be a function, which can calculate the distance between the outputs of the fully connected layers. Examples of important transfer learning loss functions are described in Section 2.4.6. In addition, there are still the normal loss functions of the CNN training process L_{Source} and L_{Target}, which have been defined as cross-entropy in the previous section. This leads to a total loss function for the transfer learning process L_{Total}, which is the sum of all loss functions: L_{Source}, L_{Target}, L_{FC1} and L_{FC2}. As shown in Eq.(20), each of them is used with its tradeoff parameter λ to modify the intensity of the respective loss function. This follows the proposal of the usage of λ for domain adaptions, as introduced in Section 2.4.4 for the use case of an MMD loss function.

$$L_{Total} = \lambda_{Source} * L_{Source} + \lambda_{Target} * L_{Target} + \lambda_{FC1} * L_{FC1} + \lambda_{FC2} * L_{FC2} \tag{20}$$

L_{Source} is the loss of the labeled source domain data and is used in combination with its tradeoff parameter λ_{Source}. Accordingly, L_{Target} is the loss of the labeled target domain data and its corresponding tradeoff parameter λ_{Target}. The loss functions L_{FC1} and L_{FC2} with their tradeoff parameters λ_{FC1} and λ_{FC2} are used for the fully connected layers 1 and 2. They are calculated with the output of their fully connected layers and with the help of a transfer function like MMD. Since the mentioned loss functions for $L_{FC1/2}$ do not need any label information, unlabeled target data can be used.

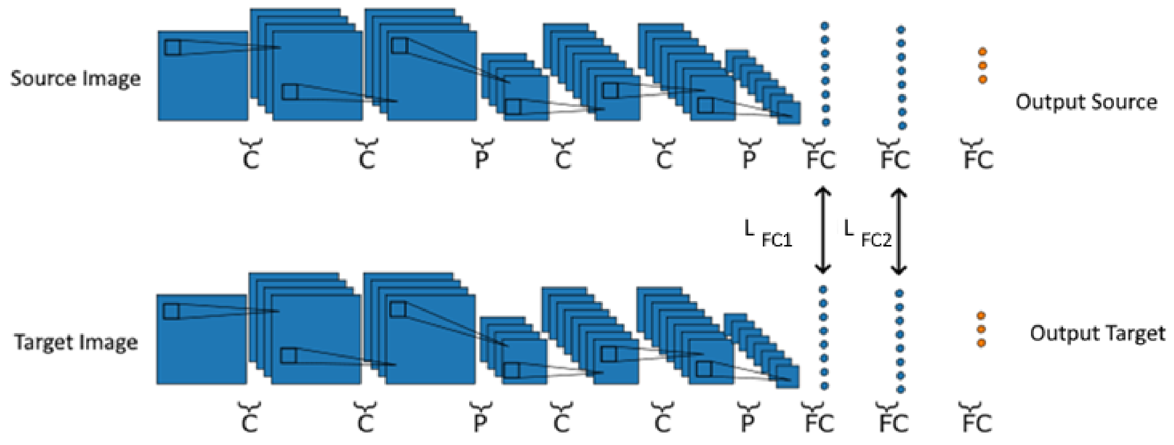

Figure 40: Simplified representation of the proposed CNN layout for transfer learning. It is a mirrored model that shares the same frozen weights and biases in the convolutional layers. Only the fully connected layers are trained during transfer learning with the help of the original loss function plus additional transfer learning losses L_{FC1} and L_{FC2} [133].

The above-defined transfer learning approach is referred to as TLA1 in this thesis. Using Eq. (20), it can be used for a semi-supervised transfer learning approach as well as for unsupervised transfer learning. Therefore, it is universal. In the unsupervised case, there are no labels available, which results in L_{Target} also not being available. Since, in this case, $\lambda_{Target} * L_{Target}$ is zero, Eq. (20) can be simplified to Eq. (21).

$$L_{Total\_unsupervised} = \lambda_{Source} * L_{Source} + \lambda_{FC1} * L_{FC1} + \lambda_{FC2} * L_{FC2} \tag{21}$$

5.4.4 Alternative Transfer Learning Approach

For the second semi-supervised transfer learning approach (TLA2), the CNN architecture remains the same as for TLA1. Only the process has to be adapted. Therefore, the first step is equal to TLA1 when doing unsupervised transfer learning with unlabeled data: first, the CNN must be pre-trained with labeled source data. As shown in Figure 41, this is followed by a transfer learning step. This would correspond to TLA1 without any labeled target domain data using Eq. (21). Here, the convolutional layers are frozen as described above, and the unlabeled part of the target data is used together with the labeled data of the source domain to train the network. The last step uses the labeled target domain data. The pre-trained model, which still has frozen convolutional layers, is now trained supervised with the remaining labeled target data. This reduces the loss function of Eq.(20) to Eq. (22) during this transfer learning step.

$$L_{Total\_target\_supervised} = \lambda_{Target} * L_{Target} \tag{22}$$

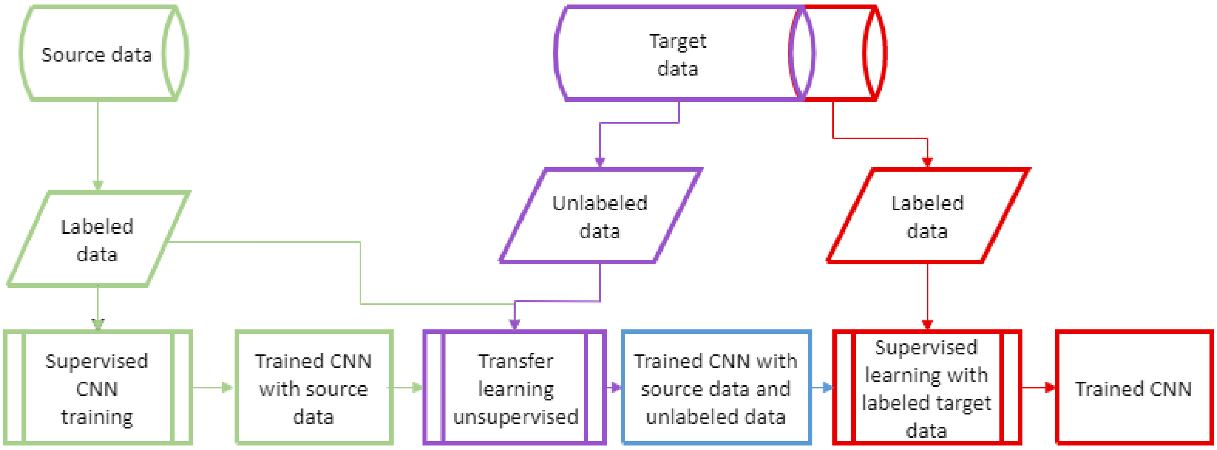
Source data
Target data
Labeled data
Unlabeled data
Labeled data
Supervised CNN training
Trained CNN with source data
Transfer learning unsupervised
Trained CNN with source data and unlabeled data
Supervised learning with labeled target data
Trained CNN

Figure 41: Alternative transfer learning process (TLA2): First, the model is trained using source data that has been labeled. Afterward, this model is trained once more using unlabeled target data. Finally, the model is fine-tuned with labeled target data (cf. [133]).

5.5 Layered Maximum Mean Discrepancy

Instead of using the existing loss functions, this thesis also introduces a new loss function for domain adaptation solutions based on CNNs like the one presented in the previous subchapter. Ordinary loss functions like those presented in Section 2.4.6 are pure data-driven approaches. By taking domain knowledge into account, existing loss functions may be improved for specific use cases [13]. In the case of CNNs, knowledge of the input image can be used. This can be, for instance, the image composition or the colors. In the context of predictive maintenance, where the input often comes from single-valued sensors, colors only make sense when the sensor data is converted into a multicolor image through a specially designed algorithm or color cameras. Since, for predictive maintenance cases, the image's content is often known, the image's composition is a promising starting point.

For the use case of components, which have different characteristic frequencies, it is obvious that a possible hybrid approach should take care of them. As can be seen in Section 2.3.4, images generated from sensor data by means of time-frequency transformations usually have the frequencies aligned along the y-axis. With this knowledge, each characteristic frequency can be seen as a separate layer, which can be optimized on its own. For this optimization, several algorithms are available. As stated in Section 3.3.4, the most promising algorithms are MMD and CORAL. Both have their advantages and disadvantages. MMD is a commonly used technique for domain adaptation with a convincing success rate. It is also less resource-intensive than techniques like CORAL. The downside of discrepancy-based approaches like MMD is that they can be vulnerable to quite different distributions of the error classes. This downside is not a big deal for transfer learning in predictive maintenance scenarios since the error distribution is often equal across different domains (see Section 3.2.4). In addition, in a test scenario, CORAL and MMD approaches have nearly the same accuracies (see Appendix A.3.4). Therefore, MMD has been selected to optimize the different domains.

Since the new loss function extends the classical MMD by using the context information of the layers, the new loss function is called Layered Maximum Mean Discrepancy (LMMD). The main difference between this hybrid approach and the original MMD (Eq. (8)), which is a purely data-driven approach, is that each layer is adapted independently. Therefore, the input images must be modified so that they represent only the features of the currently adapted layer. The features of the other layers should be ignored. To achieve this, a copy of each image is created for each layer. Then, for each copy, all pixels of the image areas that do not correspond to the analyzed layer are set to 0 (see Figure 42). Afterward, these images are used to calculate the loss of the corresponding layer with the help of the normal MMD function. Finally, the total loss function is the sum of all layer-specific loss functions and can be formulated as in Eq. (23),

$$L_{LMMD} = \sum L_i(S,T) \tag{23}$$

and Eq. (24)

$$L_i(S,T) = MMD(f(S_i), f(T_i)) \tag{24}$$

where i are the layers of the image, and $f(S_i)$ and $f(T_i)$ are the outputs of a fully connected layer associated with the corresponding layer represented by i. This loss function can be used in the CNN architecture proposed in Section 5.4 for L_{FC1} and L_{FC2}.

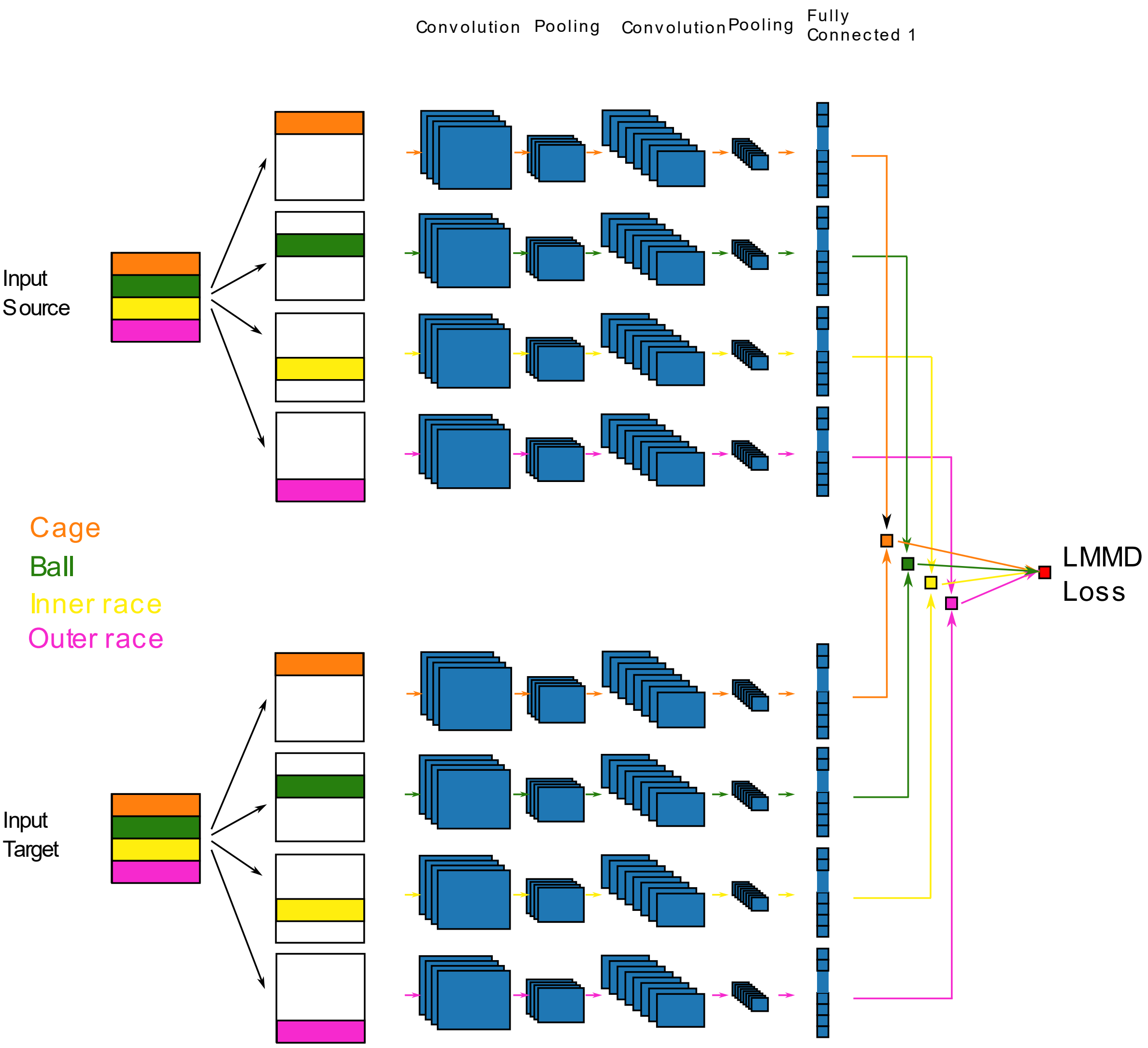

Figure 42: Calculation of the loss value with the LMMD loss function on fully connected layer 1. The image is split into separate images for each class. The output of the fully connected layer for each class is the input of an MMD function. All results together are the total LMMD loss.

The principle of this algorithm is now demonstrated using the example of the intermediate domain for bearing fault classification from Chapter 4. Here, the context information of the four frequency layers can be used. Each layer represents one fault category (inner ring, outer ring, cage, and ball fault) and consists of the first four harmonics for each category. As shown in Figure 42, the total loss function consists of four parts and is defined as in Eq. (25), which is based on Eq. (23) with i = {Outer, Inner, Cage, Ball}.

$$L_{LMMD}(S,T) = L_{Outer}(S,T) + L_{Inner}(S,T) + L_{Cage}(S,T) + L_{Ball}(S,T) \tag{25}$$

In the following section, the calculation of the loss of the layer of the cage (L_{Cage}) is explained as an example with the help of Figure 42. For this calculation, only the top quarter of the images is utilized (marked in orange in the figure). This is where the cage errors are mapped by the frequency-selective filter of the intermediate domain (see Section 4.5). All other parts are to be set to 0. Afterward, these images are used as input for the CNN to calculate the loss by the normal MMD function, which can be done after each fully connected layer. In this example, the fully connected layer 1 is used. This loss value is then summed using loss values of the inner race, outer race, and ball fault to calculate the LMMD loss value.

5.6 Generalization

The introduced transfer learning approach for classification tasks can be divided into two parts. The first is the intermediate domain presented in Chapter 4, and the second is the transfer learning approach (Section 5.4) with the help of the LMMD loss function (Section 5.5).

Although this approach was designed and evaluated for bearings, it should also be possible to use it for other components with periodic movements. This has already been explained for the intermediate domain in its corresponding generalization chapter (Section 4.8) and should also be true for the model and the LMMD loss function. Since the model itself has no dependencies on the input except the input size, a different component should also be possible. The general usability should also be given for the LMMD loss function because it is designed for the here-developed intermediate domain, which consists of different layers. Therefore, it is also suitable for intermediate domains of other components. The only thing that must be adjusted is the number of parallel layers in Eq. (25).

A different component, which may be handled with the presented classification approach, is, for instance, a gear. Gear fault classifications are also carried out based on fault frequencies. Here, the characteristic frequencies are the gear rotational frequency, the pinion rotational frequency, and the gear mesh frequency [172]. Therefore, some modifications must be made. At the very least, the number of layers used for the intermediate domain must be changed, since gears have only three different fault frequencies while bearings have four different fault frequencies. This leads to having only three layers in the intermediate domain, which necessitates changing the number of parallel layers used for the LMMD function in Eq. (25) to three. This assumption is based on the assumption that gears may have different fault frequencies and different manufacturing tolerances. The hypothesis of the adaptability to gears is based on theoretical considerations and has not been verified based on a reference data set.

5.7 Conclusion

This chapter presented a new predictive maintenance solution for transfer learning of classification tasks for all sorts of components, which have characteristic frequencies such as rotors, gears, or bearings. This approach was demonstrated in detail using the example of bearing fault classification. However, as elaborated in Section 5.6, it is a generic approach and should fit for the previously mentioned components as well since there is no bearing-specific algorithm used.

To summarize, the presented solution uses the new intermediate domain of Chapter 4, which is specially designed for the classification task of characteristic frequencies, which takes their unique features in the form of layers into account. This intermediate domain can be applied afterward as input of a CNN. Therefore, a double-layered convolutional neural network was presented. Furthermore, a new loss function for domain adaptation (LMMD) that considers the particular layout of images, which contain characteristic frequencies, has been introduced. Finally, the LMMD was explained in detail, utilizing the example of the intermediate domain for bearing fault classification. These parts present a novel predictive maintenance chain for use cases where a large, labeled source dataset exists, but the target domain consists of unlabeled or only a few labeled samples. As such, this solution can be seen as a direct answer to RQ1, which asks for a new method that provides better results for fault classification of partly labeled bearing fault datasets. The verification of this classification approach is provided later in Section 7.2 in the form of an exploration, where it provides an up to 32.3% improved accuracy in contrast to other state-of-the-art approaches. Furthermore, a benchmark in Section 7.3 shows that the given approach delivers better results than the benchmarked approach.

In addition to answering RQ1, the presented classification approach also provides a part of the answer to RC3, which asks how existing methods can be combined and optimized. The answer is the solution itself, which combines and optimizes existing approaches like time-frequency transform, CNN, and domain adaption into an approach that provides better results for fault classification of partly labeled datasets. In addition, LMMD is an optimization of the MMD algorithm that provides better results for the given use case.

6 Transfer Learning Approach for Remaining Useful Life

6.1 Introduction

The current state of the art for predictive maintenance solutions in general (Chapter 2 and 3) and remaining useful life for bearings in particular (Section 3.3.3) have shown that, currently, most approaches focus on RUL without transfer learning or on transfer learning between different conditions of the same type of component. Today, only two published research works benefit from datasets of different component types via transfer learning. Such a component can be, for instance, a bearing, as already described in the classification approach in Chapter 5. In addition, the RUL estimation and the classification task suffer from a lack of reasonable training data. For the RUL estimation, the lack of training samples is even more significant than for the classification task. The whole lifetime of a component must be tracked in order to train a neural network, and it is not sufficient to measure the component once a fault appears. Because of this, knowledge transfer promises to be important, especially for new types where very little training data is available. An additional challenge arises from the fact that the sensors are permanently mounted for continuous RUL estimation. This contrasts with the classification task, where the use cases only need a temporarily attached sensor to check the current state of the component. Due to the cost pressure in industrial applications, the most cost-effective RUL solutions must be used. This can be realized by using permanently mounted triaxial sensors, which, however, have the disadvantage that they can only cover a limited frequency range of up to approx. 5 kHz (see Section 3.2.5). As stated in Section 3.3.3.6, there is no current solution that covers these needs.

To overcome these shortcomings, an RUL solution that meets the following requirements is needed:

- The solution shall be a general solution that can be applied to different machine components. These components must meet the criterion that they are moving parts with rotational movements.
- The failure of such a component shall not be spontaneous without indication but must become apparent in some continuous trend. This requirement is essential because if the state of a component changes because of unforeseen conditions, it is not possible to detect the state correctly [177].
- A frequency range of a maximum of 5 kHz should be used to use triaxial accelerometers, which are needed to realize cost-efficient industrial applications.
- It shall have superior prediction accuracy compared to current state-of-the-art techniques, especially for real-world scenarios. Therefore, existing approaches can be combined suitably. In addition, the usage of domain knowledge can also be an opportunity.

- In order to overcome the lack of existing training samples for new component types and process conditions, it is essential to have algorithms optimized for a small number of samples.

All these generic requirements are directly applicable to the RUL estimation of bearings. In addition, a solution to these requirements mapped to bearings is also an answer to RQ2 (see Section 1.2), which asks for a new method that can take benefits of a dataset of a different bearing type, for a labeled target dataset that is recorded with sensors with low sampling rates.

A possible answer to this research question could be based on an LSTM, as LSTMs are well suited for tasks with time dependencies (see Section 2.2.5). The previous chapters have shown that the use of the proposed intermediate domain is well suited to preprocessing the input data. In such instances, the source and target domain get closer together. In order to use the intermediate domain images as input of an LSTM, the images must be preprocessed. This can be done with the convolutional layers of a CNN. Through the usage of a CNN, the mechanism of fixed feature extraction can be used to do the transfer learning between different domains (see Section 2.4.4).

This hypothesis is addressed in the solution that is presented in this chapter. It implements a complete process chain (see Figure 43) that picks up the underlying principles of the developments done in Chapter 4 and Chapter 5 for fault classification and complements them with the RUL estimation's ability to consider time dependencies. Comparing this process chain with the one presented in Figure 35 for the classification of bearing faults reveals that the main steps are equal for both tasks: The raw sensor data is used as input. Out of this input, the features must be selected for the corresponding machine learning algorithm. The machine learning algorithm is then used for classification or estimating the RUL. For the classification approach, the feature extraction uses a hybrid model based in the intermediate domain, which abstracts the input data to focus only on the relevant frequency areas (Chapter 4). In addition, the convolutional layers of the CNN are also used for a data-driven machine learning-based feature extraction. For the classification, only the last fully connected layers are considered. This hybrid feature extraction can be taken over by the RUL task (Section 6.3) because the data to be analyzed is identical. In contrast to the classification task, the RUL task does not need to assign the input data to different conditions. Instead, it must estimate a time. Therefore, the output of the feature extraction based on the convolutional layers of the CNN is used as input of an LSTM network (Section 6.4), which is used to estimate the RUL with the help of a health indicator (see Section 3.3.3.1). In addition, transfer learning to different other bearing types can be performed with a fixed feature extraction (Section 6.4.4). Each section explains its design decisions as a first step and then explains them in detail.

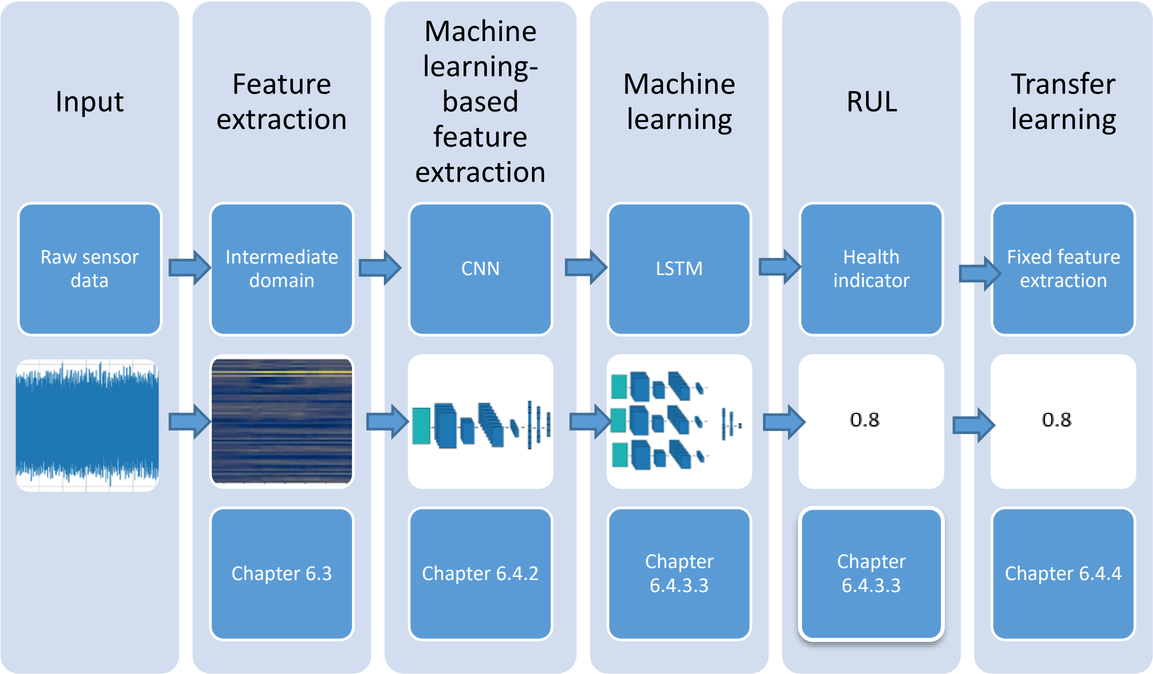

Figure 43: The different steps for a transfer learning-based RUL estimation approach. The input, which is the raw sensor data, is preprocessed with a feature extraction technique, such as an intermediate domain. Afterward, the extracted features are used for the machine learning process, which is based on a CNN in combination with an LSTM. The CNN can be pre-trained with data from a different domain for the feature extraction and then transferred to a learning task in the target domain. The trained model is finally used for the regression task of the RUL estimation with the help of a health indicator (cf. [112]).

The presented approach is verified in detail with a benchmark in Section 7.4 of this thesis. The presented RUL solution has been summarized in a research paper [112].

6.2 Validation Context

All design decisions in this chapter are based on empirical test scenarios based on bearing datasets. Each test scenario is based on two datasets. First, the feature extraction part of the proposed RUL is pretrained with the Case Western Reserve University dataset (see Section 3.3.2.2) and the mean accuracy in the same way as already described in Section 5.2. Every test scenario uses these pretrained convolutional layers.

For the RUL estimation, the dataset of the IEEE PHM 2012 data challenge is used. This dataset, which was introduced in Section 3.3.3.2 is the most widely used RUL dataset. The different datasets were assigned according to the contest specification, where six training datasets and 11 test datasets were used [178]. A suitable metric has to be selected to verify the different development steps. A common metric to compare the accuracy of different approaches is to use a particular score, which is calculated by summing up the weighted relative error rate of each test case (see Appendix A.5.1). This score is also used to evaluate the different approaches in this chapter. Furthermore, this score was also used to compare the different solutions of the IEEE PHM 2012 Data Challenge [178]. For this reason, it is also referred to as the PHM score within this thesis. The score of each test scenario was the decision basis for each evaluated parameter.

The parameters of the training itself are listed in Table 12

Table 12: Training parameters used for the proposed RUL approach.

Parameter	Value	Reason
Optimizer	Adam	Empirical tests and recommendations, such as [6].
Learning rate	0.0005	Empirical tests.
Batch size	120	Due to the limitations of the used hardware, a larger batch size could not be used.
Epochs	300	An improvement of the result could not be achieved with more than 300 epochs.
Loss function	Mean squared error	MSE was used based on the recommendation of Liu et al. [179] that, amongst all common loss functions, it is the most sensitive one to measurement errors.

6.3 Intermediate Domain

As explained in the introduction, the estimation of the RUL starts with the preprocessing of the sensor-based raw input data. According to Section 4.3, the acquisition of the input data is equal for the task of the classification of the health state and the estimation of the RUL. In both cases, a sensor collects information (e.g., acceleration) for a short recording phase of the length *t.* This can be done during specialized test runs. This recording is repeated periodically at an interval *T* to assess the actual conditions of the component. It is sufficient to analyze only the last record for the classification task. However, the RUL task can also benefit from analyzing historical data of previous recording phases to consider time dependencies. In accordance with the literature presented in Section 3.3.3, there are a number of different feature extraction techniques, such as using different time and frequency domain features or wavelet transformations. However, the intermediate domain in Chapter 4 is specially designed and validated for this kind of sensor data. This method has already been used successfully for classification tasks in Chapter 5. In addition, the use of the intermediate domain provides the advantage of supporting transfer learning, since it already brings the data of the target and source domain closer together. For these reasons, this method is also considered to be the most suitable feature extraction method for the RUL estimation.

Using the intermediate domain is also important for the requirement of having a solution that is usable in combination with triaxial sensors. Since the intermediate domain only focuses on the characteristic fault frequencies of a component, other higher frequencies are not used. It is even possible for components with high rotation speeds to have frequencies (including the first four harmonics) below 5 kHz, which is in the range of triaxial sensors. An extreme example of such a component is a bearing of a high-speed grinding spindle, which can have rotational speeds up to 23,000 rpm. This results in fault frequencies up to 4,500 Hz when using the bearing at 100% of the maximum rpm [133]. The maximum frequencies of these fault frequencies can be reduced by the aforementioned test runs, in which the spindle has to rotate at specified rotational speeds. If such a high-speed spindle is used at 10% of the maximum rpm, it will result in fault frequencies of only about 500 Hz. Bearings for other use cases have rotational speeds of about 2,000 rpm, resulting in fault frequencies of less than 200 Hz [133].

In summary, it can be said that even for the extreme case of bearings in high-speed spindles, the requirement of usability with triaxial sensors can be fulfilled by specialized test runs that are not carried out with the maximum speed of the spindles. When using simpler bearings applications with lower rotational speeds, it should not be necessary to pay attention to the rotational speeds.

One constraint of the usage of the intermediate domain is that it relies on the characteristic fault frequencies. For the use case of bearings, the first appearance of these frequencies is at degradation stage 3 (see Section 3.2.4). Therefore, this approach is unable to recognize an incipient bearing damage in fault stage 2. However, this limitation is also based on the triaxial sensors used. In stage 2, only natural frequencies of the bearing components occur, which are up to 6 kHz and therefore not completely covered by these sensors. However, as described in Section 3.2.4, this is not a real-life problem. In stage 3, there is still enough time for a planned maintenance. In addition, other current predictive maintenance solutions also rely only on stage 3.

6.4 Proposed RUL Architecture

6.4.1 Introduction

The preprocessed images of the intermediate domain can be used as input for a machine learning algorithm. As introduced in Section 6.1 these images must be processed to use them for the RUL estimation. Therefore, features must be extracted from the images. Afterward, these features must be used for the RUL estimation. As stated in Section 3.3.3, a common technique for regression tasks such as RUL is an RNN. In addition, LSTMs are also important due to their ability to optimize the vanishing gradient problem of RNNs (see Section 2.2.5). Different surveys have compared the relevant machine learning approaches in the context of estimating the RUL for bearings [54, 132, 180]. They outline that the current RUL approaches are based on different techniques, such as deep learning techniques like CNN and RNN, and classical machine learning techniques like SVR and Bayesian Monte Carlo. These techniques are mainly used for supervised learning tasks in the same domain. CNN-based approaches [12], which use the raw sensor data, and LSTM-based approaches [82], which use extracted features out of the time and the frequency domain, are used for transfer learning between different domains (see also Section 3.3.3.4).

A promising approach seems to be the combination of CNN and LSTM, where the CNN is used for the feature extraction of the intermediate domain images. This approach is already used for different types of machine learning based sensor diagnostics. Including e.g. sensors in the field of automotive sensor systems [32], but also the in this case important field of RUL estimation for bearings (see Section 3.3.3.5). The LSTM can use these extracted features of the CNN and also take the time dependencies of the different measurements into account to estimate the RUL. Therefore, this approach would be based on two feature extraction layers: The first layer extracts the features of the raw sensor data with the help of a frequency-selective filter to intermediate domain images. As illustrated in Figure 43, the second feature extraction layer uses the intermediate domain images as input for a CNN. Its output is used as input for an LSTM, which performs the RUL estimation. This design approach is explained and derived in this chapter.

6.4.2 Machine Learning-Based Feature Extraction

A common method for feeding an LSTM or an RNN is to directly use the raw signal or extract features based on statistical features of the time or frequency domain (see Section 3.3.3). As already justified in Section 6.3, this RUL approach uses the intermediate domain of Chapter 4 to preprocess the raw sensor signal into a 2D image. To use images as input for machine learning methods, a feature extraction is recommended to reduce the training time and increase the precision of a trained model [158, 181]. Based on the theoretical background in Section 2.2.4 and the practical demonstrations of different research papers in Section 3.3.3.5, a machine learning-based feature extraction is appropriate to effectively and automatically extract features. A suitable technique for the feature extraction of multidimensional input is to use the convolutional layers of a CNN [181]. As introduced in Section 2.2.4, a CNN uses the lower convolutional layers for low-level feature extraction and the higher convolutional layers for high-level feature extraction. Only the fully connected layers are used for the classification task.

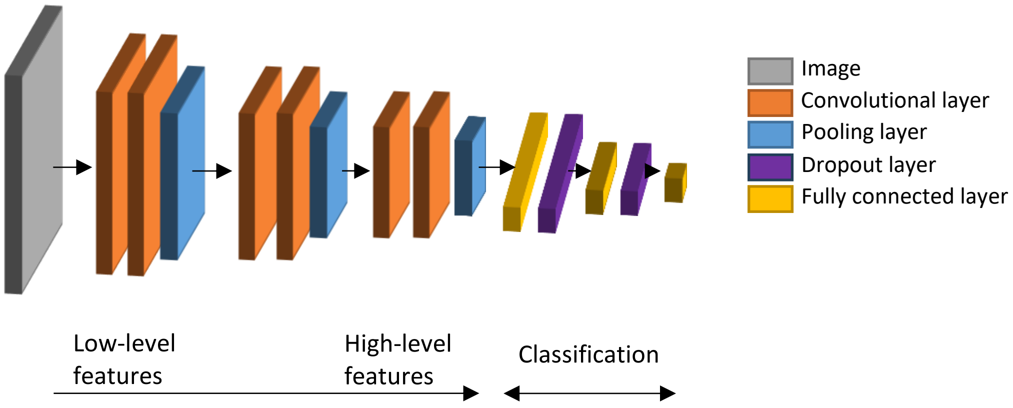

Figure 44: Feature extraction and classification parts of a CNN. The features start from a low-level on the left side and end at a high-level on the right side [112].

Chapter 5.4.2 suggests the proposed double-layer CNN architecture as an appropriate approach for feature extraction of the intermediate domain images for the classification task. As shown in Figure 44, it is based on three blocks of two convolutional layers followed by a pooling layer for the feature extraction. These blocks are followed by three fully connected layers, where the last layer is responsible for the final classification of a specific health state. Since the convolutional layers are responsible for detecting local conjunctions of features of the previous layer and mapping them into a feature map, they can be used to extract the features from the intermediate domain image. Therefore, the whole feature extraction part of the classification model (all convolutional layers and pooling layers) is also suited for the RUL approach. It uses these layers as a machine learning-based feature extraction method to extract the features of the intermediate domain in a suitable way and passes them on to the LSTM network.

6.4.3 Time dependencies

6.4.3.1 Introduction

The extracted features of the previous step can now be used to estimate the RUL. Therefore, this chapter presents an LSTM model. First, a suitable degradation model is chosen in Section 6.4.3.2. This is followed by the introduction of a proposed LSTM model for the RUL estimation in Section 6.4.3.3. As the last step, an investigation of the window size of the inputs is presented in Section 6.4.3.4.

6.4.3.2 Degradation

In addition to the ML model itself, the degradation model is also an important decision. The most common ones are a linear degradation model and a piece-wise linear degradation model [182]. The linear degradation model is used in numerous research works with different deep learning techniques such as CNN, RNN, and LSTM (see 3.3.3.3 and 3.3.3.4). For this case, the degradation is considered linear over the entire lifetime of the component (see Figure 45a).

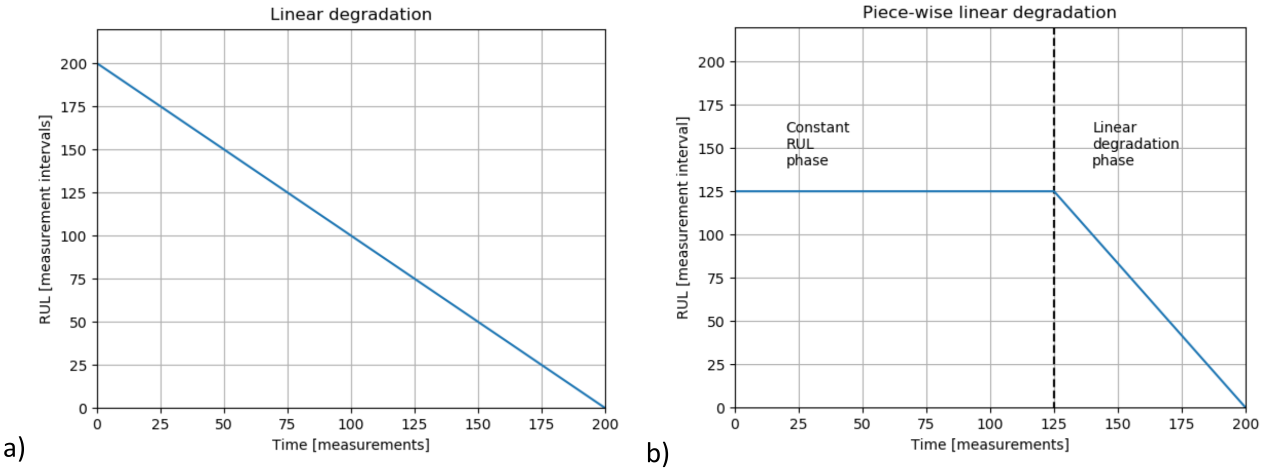

Figure 45: Different degradation models for an example with a total lifetime of 200 measurements. a) shows a linear model from start to end of the lifetime. b) shows a piece-wise linear degradation model, where only a degradation phase exists at the last n=75 measurements. For the other measurements, no degradation can be determined.

Instead of linear degradation, a piece-wise linear degradation may also be an option. When using a piece-wise linear degradation, only the last *n* values are treated as a linear degradation of the component. The degradation of all other measurements is always considered as constant (see Figure 45b). This approach can be used in cases where once the degradation has started, it always takes about the same time until the component reaches the end of life. However, this approach has two drawbacks [182]: first, a maximum estimated RUL is limited to the chosen length *n* of the degradation phase. In addition, the more severe drawback is that the degradation phase is not comparable between different samples in many use cases. This is a significant problem for many predictive maintenance solutions. Some researchers have attempted to circumvent this problem by dynamically calculating the end of the constant RUL phase for each bearing. This can be done using different statistical functions that are applied to the sensor values. As soon as a specific threshold value is reached, the constant RUL time is declared as terminated. For this purpose, however, these threshold values must be determined empirically for each bearing type [183].

Some approaches do not use a linear health indicator. For instance, different statistical features from the time and frequency domain data can be used as it is done, for instance, by Khan et al. [184]. However, such an approach again leads to the problem that threshold values for the individual features and the optimal combinations must be determined. Khan et al. used specially developed MATLAB programs for this purpose.

The research directions for a dynamic constant RUL phase end and an individual health indicator contradict the approach of having a universal approach that can be simply applied to other bearing types. Therefore, even though they may provide better results for a particular bearing type, they are not considered further here.

The general choice between using a simple linear health indicator and the piece-wise linear degradation remains. As shown in chapter 3.2.4, the degradation of the same bearing type is very similar in most cases. Especially the degradation levels, which can be detected with the relevant accelerometers (degradation stages 3 and 4), are for 90% of all bearings of one type in a range of 5% of the expected theoretical lifetime. This can also be seen in Figure 46, where an exemplary bearing of the dataset presented in Section 3.3.3.2 is shown. Therefore, there is a range and no exact value when the constant RUL phase of the piece-wise linear degradation is over. This results in no advantages in the usage of a piece-wise linear degradation. Another downside of the piece-wise linear degradation is that the length of the constant RUL phase must be determined somehow for each bearing type. This again leads to an approach that requires the manual intervention of experts.

For these reasons, the typical approach of only using a linear degradation over the entire lifetime has been chosen for the presented solution. In addition, the literature also recommends using a linear model if the knowledge of a suitable degradation is unavailable [182].

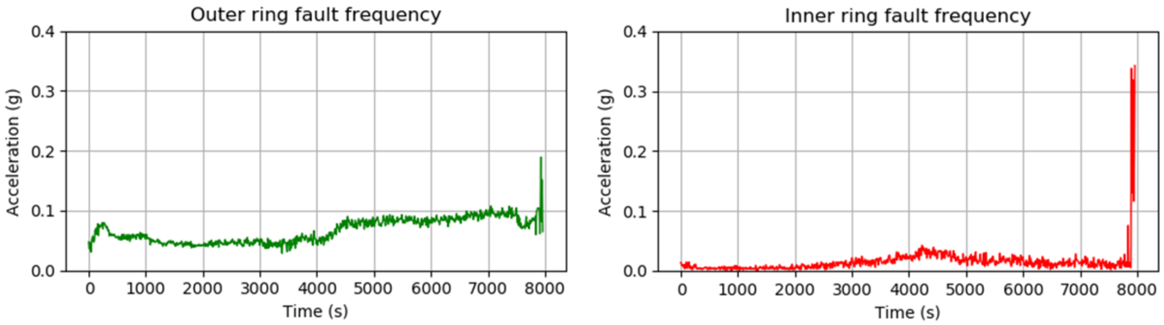

Figure 46: Example of the degradation of a bearing based on the outer and the inner ring fault frequency.

6.4.3.3 LSTM Model

One crucial aspect of an RUL solution is the part that uses the features of the input data to estimate the estimated RUL. This is because, as presented in Section 3.3.3 through different research, the accuracy of an RUL solution depends significantly on this part. As outlined in Section 6.1, this dissertation searches for an appropriate technology for this purpose, which can also take advantage of available historical data. As outlined in Section 3.2.6, deep learning technologies are proposed for complex scenarios such as RUL estimations. These technologies are similar in that they are typically trained with normalized data in a range between 0 and 1 (see Section 4.6). Therefore, as mentioned in Section 3.3.2.1, the deep learning-based RUL approaches use a health indicator, which is between 1.0 for a new component and 0.0 for a defective component. As proposed in Section 6.4.3.2, the interpolation between 1.0 and 0.0 is linear over the entire lifetime. This health indicator has to be retransformed in a separate step back to a time span.

The RUL estimation through a health indicator is realized in deep learning approaches by a single fully connected neuron as the last layer of the neural network. This is done in different studies with different deep learning approaches such as CNN, RNN, and LSTM (see Sections 3.3.3.3 and 3.3.3.4). However, a CNN is not suitable to represent a time relation because only one input is analyzed at a time, and there is no connection to the prior inputs. By considering time relations, higher accuracies might be reached (see Section 2.2.5). Unlike CNNs, RNNs, and their successors GRUs and LSTMs, use data from previous time points. As already introduced in Section 2.2.5, the disadvantage of an RNN is that it is sensitive to the gradient vanishing problem. This problem can be mitigated by means of an LSTM or a GRU architecture. In addition, a comparison between GRU and LSTM shows that their accuracy is very similar [185]. For the use case of bearings, it has been shown that an LSTM can deliver higher accuracies than a GRU [186]. Different surveys pointed out that there are also a number of successful applications using LSTMs in the field of RUL of bearings [54, 131]. Consequently, the proposed model is based on an LSTM network rather than an RNN or a GRU.

In order to choose a suitable layout for the LSTM network, an LSTM layout of Sahoo [30] is used as a starting point. In his work, Sahoo uses three LSTM layers (layer 1: 128 outputs, layer 2: 64 outputs, and layer 3: 32 outputs). The output of these LSTM layers is processed by fully connected layers for the RUL estimation based on the last fully connected layer with a single neuron. In his scenario, this implementation provides remarkable results for the task of the RUL estimation.

Based on the proposed LSTM layout, the three modifications presented in Table 13 are evaluated. All layouts have the first three layers in common: a CNN for the feature extraction, an LSTM with 128 outputs, and an LSTM with 64 outputs. The other subsequent layers are different.

- Layout 1: This layout investigates a pure LSTM network. Therefore, the CNN-based feature extraction together with two LSTM layers is used, followed by an LSTM with 32 outputs in the last layer as in the reference layout proposed by Sahoo [30]. The outputs of the LSTM are directly used as input for the health indicator, which consists of a fully connected layer with one neuron.
- Layout 2: This layout investigates the common use of one LSTM followed by several (deep) fully connected layers [187]. For this purpose, two additional layers are added to layout 1: After the last LSTM layer, a fully connected layer with 32 outputs is inserted. This layer is followed by a dropout layer with a dropout rate of 0.5. The decision to use 32 neurons was inspired by the success of double convolutional layers for CNNs, where the same output size is used for two consecutive layers (see Section 5.4.2.2). The dropout factor of 0.5 is based on the research of the classification approach in Section 5.4.2.3, as well as on recommendations in the literature [6].
- Layout 3: This layout is similar to layout 2, except that the last LSTM layer with 32 outputs is unavailable. This layout has been selected to investigate the influence of the LSTM layers. If this approach shows the best results, a simpler model with fewer layers could be used.

Table 13: The different evaluated LSTM layouts. The number of outputs for each layer is given in parentheses. Only the used layers and their connection (marked by an arrow) are drawn for each layout. For comparison, the PHM score of each layout for the RUL task of the IEEE PHM 2012 Data Challenge has been calculated. Layout 2 has the best results.

	Layers	Layout 1	Layout 2	Layout 3
Layout	CNN (8192)	🟧	🟧	🟧
	LSTM (128)	🟩	🟩	🟩
	LSTM (64)	🟩	🟩	🟩
	LSTM (32)	🟩	🟩	↓
	Fully connected (32)	↓	🟨	🟨
	Dropout (rate=0.5)	↓	🟪	🟪
	Fully connected (1)	🟨	🟨	🟨
PHM score		0.1094	0.35	0.05647

As shown in Table 13, the best result in terms of the PHM score is achieved with layout 2. Therefore, layout 2 has been chosen as the LSTM layout. A description of the evaluation can be found in Appendix A.5.2.

A detailed summary of the complete proposed model with its 4,323,265 trainable parameters is shown in Table 14. As shown in Figure 47, for each of the *n* input images, the same CNN part with the same weights is used. The output of each of the *n* CNNs is used as input for the first LSTM layer. The last fully connected layer is used to calculate the health indicator. Based on the decision of the used network and the used linear degradation model, this indicator has to be converted back to a time value according to Eq. (26),

$$T_{RUL} = \frac{T_{Cur}}{1 - HI} - T_{Cur} \qquad (26)$$

where T_{Cur} is the current time stamp, and HI is the health indicator.

Table 14: This table shows the detailed architecture of the proposed RUL framework with all used layers and their parameters. Therefore, each layer is listed with its type, the output shape of each layer (none has to be replaced with the number of images that are used in a batch and is therefore dependent on the training parameter batch size) the number of trainable parameters, and the activation functions used. The parameter w in the output shape is for the window size of historical data.

Layer	Type	Output shape	Trainable parameters	Activation
0	InputLayer	(None, w, 64, 64, 1)	0	
1	Conv2D	(None, w, 64, 64, 32)	608	ReLu
2	Conv2D	(None, w, 64, 64, 32)	9248	ReLu
3	MaxPooling2D	(None, w, 32, 32, 32)	0	
4	Conv2D	(None, w, 32, 32, 64)	18496	ReLu
5	Conv2D	(None, w, 32, 32, 64)	36928	ReLu

6	MaxPooling2D	(None, w, 16, 16, 64)	0	
7	Conv2D	(None, w, 16, 16, 128)	73856	ReLu
8	Conv2D	(None, w, 16, 16, 128)	147584	ReLu
9	MaxPooling2D	(None, w, 8, 8, 128)	0	
10	Flatten	(None, w, 8192)	0	
11	LSTM	(None, w, 128)	4260352	tanh
12	LSTM	(None, w, 64)	49408	tanh
13	LSTM	(None, w, 32)	12416	tanh
14	Fully connected	(None, 32)	1056	ReLu
15	Dropout (factor 0.5)	(None, 32)	0	
16	Fully connected	(None, 1)	33	Linear

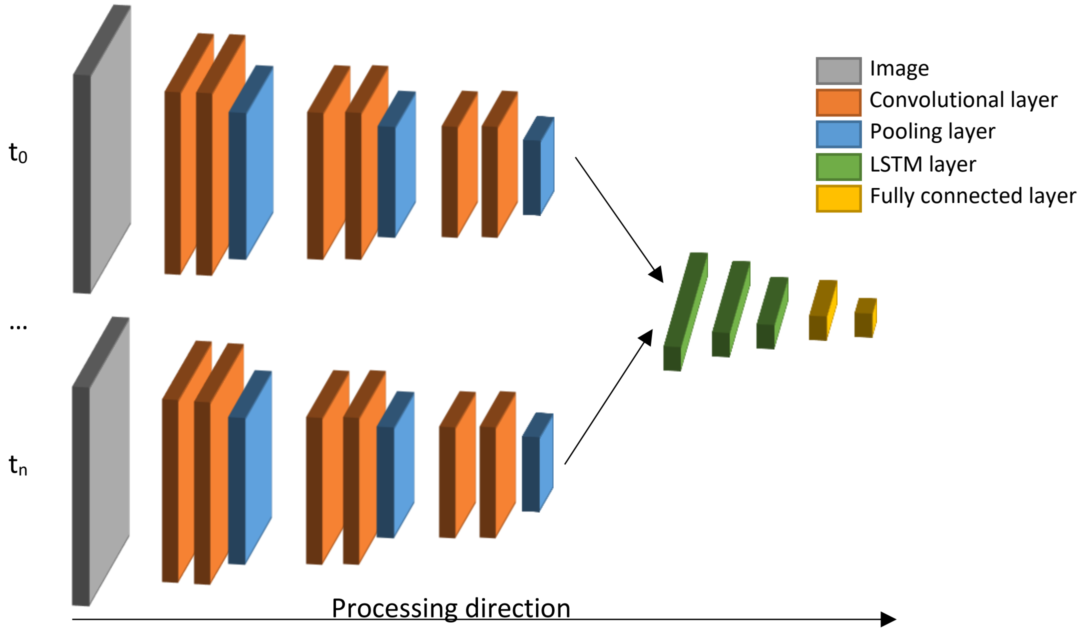

Figure 47: Proposed RUL framework. The feature extraction is performed with the help of convolutional and pooling layers. This part is equal for each of the n input images used. All convolutional layers share the same weights. The output of the last pooling layer of each image path is the input of the first LSTM network. Three LSTM layers are used. After the last LSTM, two fully connected layers are used. The output of the last layer is the health indicator of the current lifetime.

6.4.3.4 Window Size of Measurements

When using an LSTM, an important parameter is the window size *w* of the historical data used. In the context of predictive maintenance, this indicates how many measurements are included in the RUL estimation. The fewer measurements used, the more difficult it is to reflect time dependencies. However, a too-large window size can lead to an RUL determination not being possible at all. This is because if a window size *w* is defined, the RUL of a component can be determined only if at least *w* measurements are available. The neural network expects input in the selected window size only.

Another difficulty is the amount of memory needed when more historical data should be used. As shown in Figure 48, the amount of memory used during the training only has linear growth, but it quickly exceeds 16 GB. 16 GB is the amount of memory available on current mainstream graphic cards for commercial clouds (e.g., an NVIDIA P100 [65]).

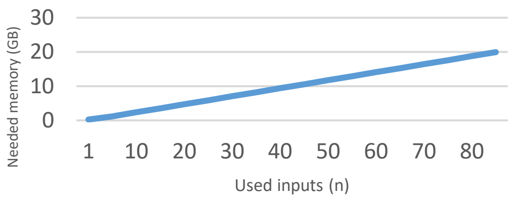

Figure 48: The memory used for the proposed LSTM network depends on the number of input images used.

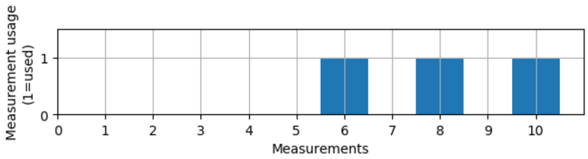

Figure 49: Detailed example of the window size of the input. A total number of 10 measurements, a step size of s=2 and an amount of used septs n=3 leads to the usage of measurements 6, 8, and 10 as input.

Regardless of these limitations, an ideal length can be determined empirically in the applicable range. One way to increase the window size without increasing the input size *n* of the LSTM network is to skip values by using a step size *s.* For instance, as shown in Figure 49, if *s* = 2, every second measurement is skipped. This results in the possibility of analyzing a time span twice as long with the same memory usage. However, skipping values has the risk that relevant dependencies are also dropped. An example is the case when only two values should be used. If only the first and last value of a measurement series are used, a wide range would be covered, but the information is not available shortly before the current time. Therefore, it could be better if no points were skipped and only the last two points were used, which would make the gradient of the values at the current position more apparent. In addition, it could be that no changes in the input values occur at all in the starting area for a long period of time.

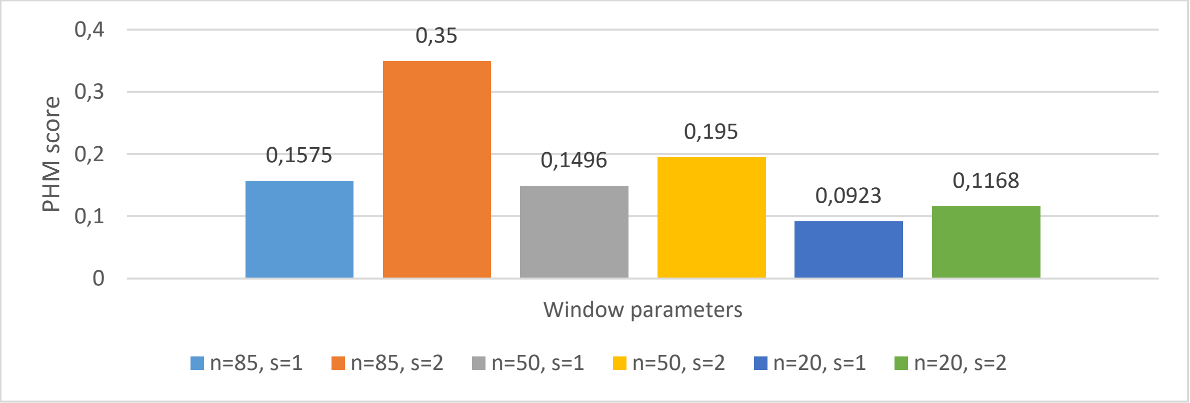

Figure 50: Evaluation of different input sizes (n) and different step sizes (s) for the training of the proposed LSTM framework. The best PHM score (see A.5.1) is attained with an input size of 85 and a step size of 2 [112].

The proper values for the parameters *n* (input size) and *s* (step sizes) are estimated by an evaluation. A limitation of the dataset used for this evaluation is that the shortest test sequence consists of only 172 measurements. Therefore, the maximum usable historical data is limited to $n * s \leq 172$. This, for instance, results in a maximum value of 85 for *n* when *s* is set to 2. As shown in Figure 50, the score is always better for the same input size *n* if a step size of 2 is used, which results in a wider time range. In addition, the input size is also important. The score gets better with an increasing input size. Therefore, it can be concluded that the wider the window size *w* of the historical data, the more accurate the result will be. Using every value is not possible for reasons of memory consumption, among others, but skipping values by a step size *s,* still allows one to achieve a wide window size. This can be done automatically by knowing the available memory and the test size being covered.

For the given dataset, this is an input size of *n*=85 measurements in combination with a step size *s*=2. Here also, the best score is reached. Therefore, this setting has been chosen for the presented solution.

6.4.3.5 Conclusion

This chapter presented an LSTM-based solution for the RUL estimation, which considers time dependencies. Therefore, the variable parameters of such a solution have been identified and explained. In addition, the most suitable values for these parameters were determined based on the bearing dataset of the IEEE PHM 2012 data challenge, as described in Section 6.2. These are the degradation model, the layout of the LSTM model, and the window size in which the data is processed.

6.4.4 Transfer Learning Approach

The presented RUL framework can be used directly for supervised learning, as shown in Section 6.4.3. However, this framework can also take advantage of transfer learning in several ways. One theoretical way is to use the LMMD, which has been proposed in Section 5.5. The LMMD can be used for unsupervised or semi-supervised transfer learning. The procedure would be similar to the classification process wherein domain adaption has to be carried out after the high-level fully connected layer. However, in real-life scenarios, the RUL datasets are usually fully labeled. As described in Section 6.1, this is because these datasets can only be collected when the sensors are mounted on a machine, and the data is collected periodically. Because of the time-consuming wiring and mounting of the hardware, this is usually done during the machine setup. Therefore, in real life, the dataset usually begins with a new component. This results in every measure having a label. For this reason, unsupervised and semi-supervised approaches will not be discussed further here.

Another approach is to perform a network-based deep transfer learning, where the pre-trained feature extraction part of the CNN of a different domain is transferred to the current domain (see Section 2.4.3.5). This approach is important for applications in the real-life world. Here, often, only a few datasets for the RUL training are available for a particular component. As introduced in Section 2.4.4 and used in Section 5.4.3, a fixed-feature extraction can be used for the case of a small dataset in the target domain. In addition, the dataset of the source domain and the target domain must be similar. In the presented case, the similarity of both domains is given by the format of the images: for both domains, the images are generated with the help of the intermediate domain. This intermediate domain benefits from the fact that the input images are very similar due to the frequency-selective filter. For this reason, the lower-level feature extraction is identical, which has also been verified for the case of transfer learning for classification in Section 5.4. The entire network must be trained with the source domain dataset for this transfer learning process. Afterward, the feature extraction part, which is carried out by the convolutional layers, is frozen. Then, only the LSTM and fully connected layers are trained with the target domain dataset.

However, the transfer from one RUL dataset to another is not the only possibility. Based on the intermediate domain, which also creates similar images for classification datasets, a transfer of a feature extraction part based on classification data should also be possible. Since many publicly available classification datasets exist, this approach has special significance. In order to perform this approach, a classification network is first trained with the source data as described in Section 5.4.2. Afterward, the fully connected layers of the classification network must be removed and replaced by the parts of the proposed RUL framework, which are responsible for the RUL estimation (LSTM and fully connected layers). Finally, the convolutional layers must be frozen, and the newly inserted parts must be trained with the labeled target domain dataset. For this approach, the classification and RUL datasets do not have to be of the same component type.

A direct comparison between an RUL estimation with pre-trained convolutional layers and one without pre-training was made to confirm this statement. In addition, the pre-trained layers were from different types of a component. As can be seen in Figure 51, the pertained network has a far better PHM score. More details of the test setup are given in A.5.3.

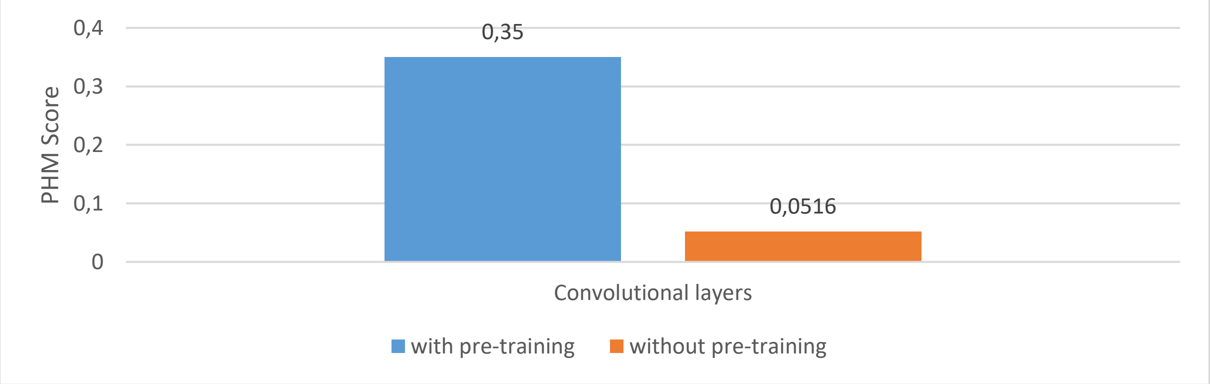

Figure 51: Comparison of the PHM score between the neuronal network with and without using pre-training convolutional layers. The pre-trained network has a much higher PHM score.

Since the results are much better with a pre-trained network, all evaluations in Section 6.4.3 were also performed with the same pre-trained convolutional layers.

6.5 Generalization

The transfer learning approach for RUL estimation consists of two parts. The first is the feature extraction, which is based on the usage of the intermediate domain presented in Chapter 4 (Section 6.3) and the convolutional layers (Section 6.4.2). The second is the RUL estimation based on an LSTM (Section 6.4.3). Both parts should be used for a range of different components.

The feature extraction part was verified in Chapter 5 for classification of sensor signals using the example of bearing fault detection based on fault frequencies. As stated in that chapter, bearings are not the only components where the intermediate domain and the convolutional layers can be applied. The underlying theory of the intermediate domain should also be applicable to other components, which can be analyzed with the help of the fault frequencies and their harmonics. If a component other than a bearing is used, parameters such as the number of used harmonics or the signal length must be adjusted, as mentioned in Section 5.6. Afterward, it can be evaluated if the component is appropriated. In contrast to the feature extraction, the LSTM model is purely data-driven and has no specific characteristic frequency-related dependencies. The only limiting condition is that the LSTM is designed to receive outputs from convolutional layers as input. Therefore, this model should also be applicable to sensor data of other components with characteristic fault frequencies. This sensor data can be converted into intermedia domain images.

A component with characteristic frequencies that might be applicable for the entire process chain of this approach, is, for instance, a gear. Here the characteristic frequencies are the gear rotational frequency, the pinion rotational frequency, and the gear mesh frequency [137]. As described in Section 5.6, some modifications must be made during the feature extraction process. Here, at minimum, the number of layers for the intermediate domain must be changed, since a gear has only three different fault frequencies while bearings have four different fault frequencies. In addition, it may be necessary to change the bandwidth of the frequency-selective filter. The RUL part might also be adapted. One possible modification might be the window size of the inputs for the LSTM.

All the above-stated assumptions about transferability to other components, especially to gears, are based on theoretical considerations and have not been verified based on a reference dataset.

6.6 Conclusion

This chapter has presented a new predictive maintenance solution for RUL estimation. This solution was specially designed and verified for the use case of bearings and consists of two parts.

The first part, feature extraction, was already presented in Chapter 4 based on an intermediate domain, especially verified for bearings but also designed for other components with characteristic frequencies. In addition, the convolutional layers of the classification approach (see Section 5.4.2) are also used for feature extraction out of the intermediate domain images. The second part is an LSTM model, which takes the output of the convolutional layers as input. This model takes the time dependencies of the degradation into account. A significant benefit of this solution is that transfer learning can be easily facilitated through the well-defined intermediate domain, which is similar for different types of the same component. Therefore, the entire feature extraction part (convolutional layers) of a pre-trained network can be transferred. This can even be a pre-trained network with datasets of a classification task of a different type.

An additional advantage of using the intermediate domain is that by using it in conjunction with special test runs (see Section 6.3), the solution can also be used for sensors with low sampling rates.

Although this solution has been verified through the example of bearings, no parts of it have bearing-specific dependencies. They only rely on periodic movements that emit periodic impulses. Therefore, this solution might be the basis for RUL solutions for other components with periodic movements. These movements result in characteristic frequencies, which are, for instance, available for rotors, gears, and bearings.

In summary, this solution answers RQ2 using an intermediate domain-based LSTM approach in combination with the proposed transfer learning approach. RQ2 asks for a new method that can take benefits of a dataset of a different bearing type for a labeled target dataset that is recorded with sensors with low sampling rates. A separate benchmark against other solutions is presented in Section 7.4 to validate the effectiveness of this solution, which is also questioned in the requirements. As elaborated in Section 6.5, this solution might also be applicable to other components with rotating elements; therefore, it might also be a solution for the general problem statement of Section 6.1. However, this assumption is based on theory only and will not be validated within this thesis.

In addition, this chapter demonstrates possible combinations of different machine learning algorithms (convolutional layers and the LSTM). This can be seen as a partial answer to RC3, which asks how existing methods can be combined and optimized.

7 Case Studies

7.1 Introduction

The previous three chapters have presented two new predictive maintenance approaches and an intermediate domain. This chapter presents case studies to validate them. First, the intermediate domain of Chapter 4 and the classification approach of Chapter 5 are validated with an exploration that compares different feature extraction methods and transfer learning loss functions. Based on this exploration, the stability of the intermediate domain parameters is also validated. Afterward, the classification approach is benchmarked against another state-of-the-art approach. Finally, the RUL approach of Chapter 6 is validated based on the well-known IEEE PHM 2012 data challenge.

In all three presented case studies, the procedures are similar:

- Source and target domain datasets are of different bearing types.
- The datasets are split bearing instance based on a training test ratio of 70:30. This has been done according to literature such as [188] and [189]. For the benchmark where a split ratio is defined, the ratio of the benchmark setup is used.
- In the first step, the neural networks are pre-trained with the source dataset. Afterward, transfer learning is performed with the target dataset or with both datasets (depending on the case study).
- The performance of the trained neural networks is measured by the accuracy in the case of classification tasks and the PHM score in the case of RUL tasks.

7.2 Exploration: Transfer Learning for Bearings Fault Classification

7.2.1 Introduction

The evaluation of the intermediate domain of Chapter 4 and the domain adaptation framework presented in Chapter 5 that is based on the intermediate domain and LMMD is performed for three different bearing vibration datasets. First, a brief description of the datasets is given in Section 7.2.2. Afterward, the evaluation is presented in Section 7.2.3. Finally, a conclusion is given in Section 7.2.4. The three case studies of this evaluation have already been published in a research article [133].

7.2.2 Data Description

Two publicly available datasets and one private dataset are used to validate the domain adaption framework. The publicly available datasets were gathered from the CWRU and the University of Paderborn, which have already been described in Section 3.3.2.2. Both contain data recorded on a test stand. The fan-end and drive-end recording positions of the data from the CWRU dataset are treated separately because they have different bearings. The dataset of the University of Paderborn is only used as data for the source domain. Therefore, only measurements with a speed of 1,500 rpm are used.

The last-used dataset is the real-life dataset of grinding spindles provided by Erwin Junker Maschinenfabrik GmbH. This dataset is also described in Section 3.3.2.2.

7.2.3 Evaluation

7.2.3.1 Case Studies Setup

The datasets described above are used in three test cases to evaluate the presented transfer learning approach of Chapter 5. To do so, three different combinations have been chosen (see Table 15). The first combination uses the CWRU dataset only. The source and the target domain are measurements from different locations with different bearing types but the same process conditions. The second uses a source dataset of the University of Paderborn and a target dataset of the CWRU. This results in different bearing types as well as different process conditions between the source and target domains. The last combination transfers knowledge from the University of Paderborn dataset to a dataset of grinding spindles. Here again, different bearing types are used in both domains. However, in contrast to case study 2, very different process conditions exist in the target domain, which are given in the form of a variance of 10% to 90% of the maximum rotational speed of the spindle.

Table 15: The different combinations of datasets used in each case study.

Case Study	Source domain	Target domain
1	CWRU drive-end side	CWRU fan-end side
2	University of Paderborn	CWRU
3	University of Paderborn	Junker Spindle

Each test case is based on the same setup to make the results comparable. The datasets of each domain (source and target domain) are split into 70% of training data and 30% of test data. This is randomly carried out four times for each case study. The presented result is the average accuracy of these four runs. Each case study uses the three health conditions: inner ring fault, outer ring fault, and healthy, which must be classified. Ball faults are not used since they are not available in all datasets. In addition, as described in Section 3.2.4, according to the literature, 90% of all faults are expected to occur at the inner and outer rings. Therefore, the not-considered ball faults are not of great importance in real life. The underlying model of the CNN is independent of the used input preprocessing method for the signal data. Three different preprocessing methods are applied to the sensor data: the intermediate domain proposed in this thesis, HHT, and S-transform. Figure 52 shows sample images of the result for each preprocessing method.

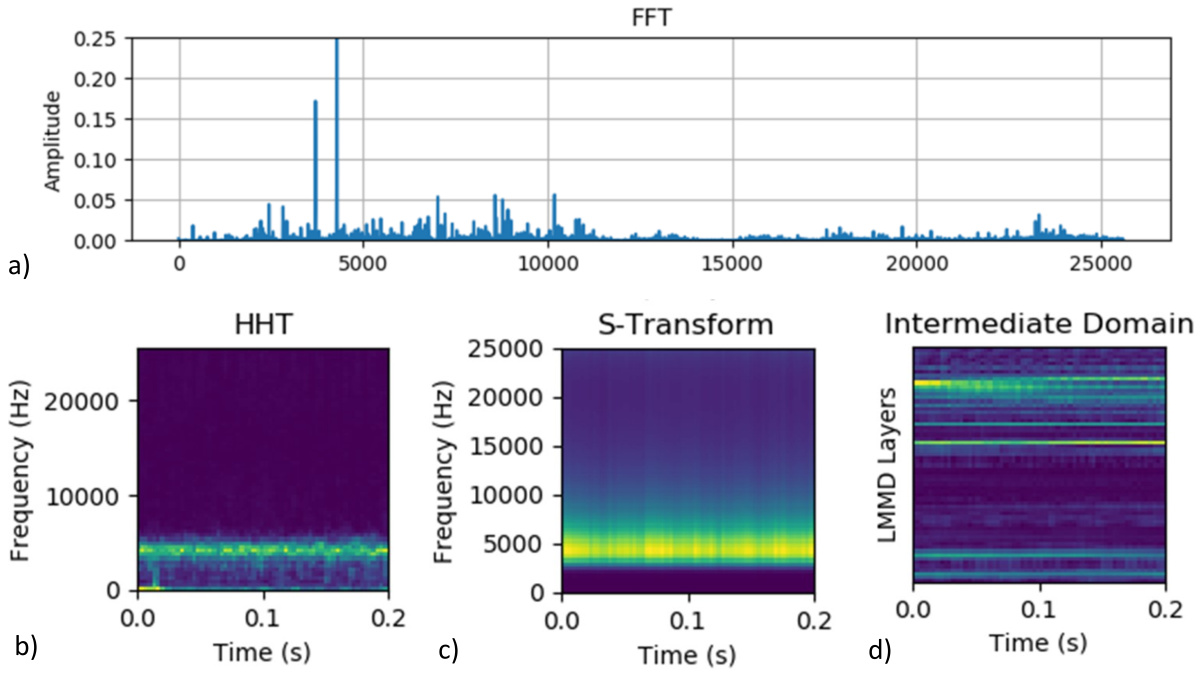

Figure 52: The results of the transformation of the same 0.2 second sample in the frequency domain (FFT) (a), time-frequency domain (HHT (b), S-transform(c)), and the new intermediate domain (d).

The transfer task itself is always performed with a CNN, as described in Section 5.4.2. That CNN is first pre-trained with the source dataset. Afterward, a transfer learning according to Section 5.4.3 and Section 5.4.4 is performed. The following setup of the transfer learning framework is used for the evaluation:

- The loss functions MMD, LMMD, MK-MMD, CORAL, and Wasserstein are used for the transfer learning process (see Section 2.4.6).
- The variable parameter γ of the Gaussian kernel function (see Eq. (9)) for MMD and LMMD is assigned as the median of the pairwise Euclidean distance of the outputs of the corresponding fully connected layers, as proposed by Ramdas et al. [190] and Sutherland et al. [191].
- The MK-MMD uses the γ values 0.01, 0.1, 0.25, 0.5, 1.0, and 2.0. This follows the use of γ by Long et al. [84].
- The tradeoff parameters λ for the fully connected layer 1 (λ_{FC1}) and the fully connected layer 2 (λ_{FC2}) are set to 50.0. This is based on an initial test of the following λ values: 0.0, 0.1, 1.0, 10.0, 50.0, 100.0 and 1,000.0. By using two exemplarily setups, Figure 53 shows that the best accuracies are obtained by λs in the range of 50. In order to indicate a general solution, 50.0 is used for all case studies.

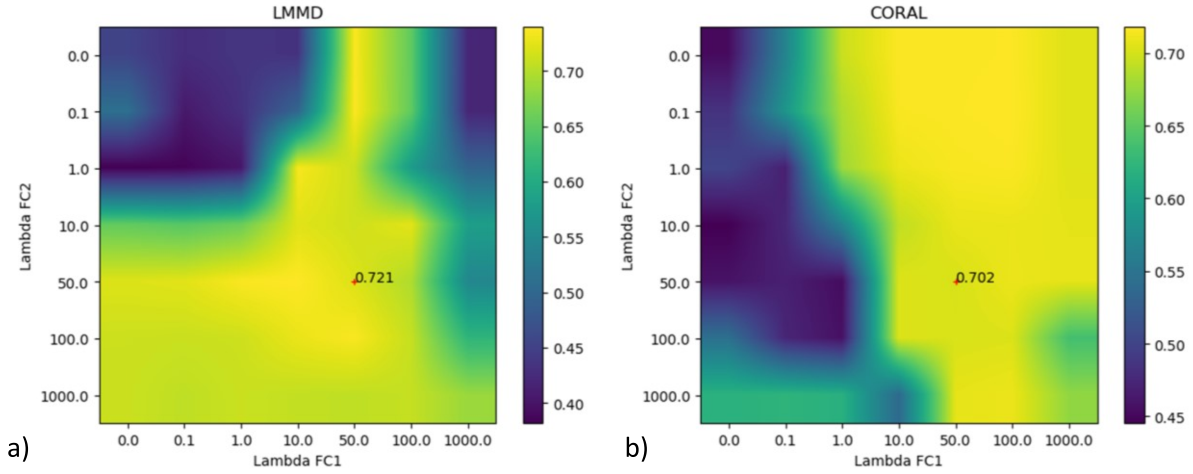

Figure 53: Accuracies for λ of 0, 0.1, 1, 10, 50, 100 and 1000 for λ_{FC1} and λ_{FC2} for a transfer learning from Uni Paderborn to CWRU using LMMD (a) and CORAL (b).

For each case study, two different settings for the training data of the target domain are used:

- Unsupervised scenario: Here, the training data of the target domain is 100% unlabeled data. Therefore, according to Eq. (20) of Section 5.4.3, only the domain adaptation approach is possible.
- Semi-supervised scenario: The target training data is divided into 20% labeled data for supervised learning and 80% unlabeled data for unsupervised learning. This split should provide insight into how well this method can be used in real-world use cases, where a lot of data is often unlabeled (see Section 5.1). Here, the following four training procedures that use transfer learning approach 1 (TLA1) from Section 5.4.3 and transfer learning approach 2 (TLA2) from Section 5.4.4 are evaluated:
    - Only unsupervised learning, according to Eq. (20) (TLA1).
    - Semi-supervised learning with labeled target data and unlabeled target data (TLA1).
    - Semi-supervised learning with labeled source and labeled as well as unlabeled target data (TLA1).
    - Only unsupervised learning, according to Eq. (20), which is followed by a pure supervised training (TLA2).

The results for both settings are presented in Table 17 through Table 22. The results refer to the accuracy of the classification of a specific fault type. Table 16 briefly describes the different headers of the tables in the semi-supervised scenario, as the tables of this scenario have more details.

Table 16: Here, the different headers/scenarios for the semi-supervised scenario are explained. They are based on only labeled data (only source domain, target, all target) and on transfer learning approaches using TLA1 (labeled data in the source, target, source and target domain) and transfer learning approach TLA2 with labeled data in both domains.

Type of transfer learning	Labeled data in	Remarks
Only source domain	**Source**	Only labeled source data is used for training; target data is used to calculate the accuracy.
TLA 1	**Source**	Pre-trained model like “only source domain.” Labeled and unlabeled source data and unlabeled target data used for transfer learning. The unlabeled source data is the same as the labeled data but without labels.
	Target	Pre-trained model like “only source domain.” Labeled and unlabeled data of the target domain and unlabeled data of source domain are used for transfer learning. The unlabeled source data is the same as the labeled data but without labels.
	Source and target	Pre-trained model like “only source domain.” Labeled and unlabeled data of both domains (source and target) are used for transfer learning. The unlabeled source data is the same as the labeled data but without labels.
TLA 2	**Labeled source and unlabeled target, followed by labeled target**	Transfer learning with transfer learning approach 2. First unsupervised like in transfer learning approach 1, “labeled source,” and then supervised with labeled target data only.
-	**Target**	Here, 20% of the training data of the labeled target domain data is used for supervised training. The model is not pre-trained with any source data.
-	**All target**	Here, 70% of the training data of the labeled target domain data is used for supervised training. As described earlier, the missing 30% of the available target domain data is used for testing. The model is not pre-trained with any source data.

7.2.3.2 Case Study 1: Classification in the Same Test Environment

The different CWRU datasets are used as source and target domain datasets for this case study. The CWRU dataset contains two parts that are both created on the same test stand on different locations (fan-end and drive-end). Both are recorded with the same loads and rotational speed. The classification process also profits from the fact that both side’s bearings are almost identical, and the faults are all artificial with same size.

1. Unsupervised-Only Scenario

The test scenario with only unsupervised data demonstrates that an accuracy of 85.4% can be achieved without applying domain adaptation by only employing the intermediate domain. This shows that the intermediate domain is stable enough to be used on these two different bearing types without modifications. In addition, it shows that transfer learning can also be carried out based on the intermediate domain for the given use case: The accuracy can be raised further, by using the LMMD technique, as well as with the MK-MMD and the classical MMD technique (LMMD 87.6%, MK-MMD 87.4%, MMD 86.4%). Another finding is that when using only source domain data, the classical HHT and S-transform approaches can achieve accuracies up to 78.3%. This accuracy can be increased up to 82.5% with transfer learning. This is caused by the quite similar source and target domains.

Table 17: Accuracies of only unsupervised learning. Source and target are different datasets of the Case Western Reserve University - Best results achieved with the help of the intermediate domain combined with LMMD (87.6%).

Setup	**Only source domain (%)**	**Domain adaption with a loss function**				
		LMMD (%)	**MMD (%)**	**Wasserstein (%)**	**CORAL (%)**	**MK-MMD (%)**
Intermediate Domain	85.4	**87.6**	86.4	50.2	80.7	87.4
HHT	68.4	-	65.5	42.5	68.8	65.3
S-transform	78.3	-	82.5	68.4	78.6	80.3

2. Semi-Supervised Scenario

The second setting of case study 1 demonstrates that employing the intermediate domain without DA produces the most accurate results with an accuracy of 93.4% when using small, labeled datasets. This indicates a negative transfer for the transfer learning cases (see Section 2.4.2), which appears for the input of the intermediate domain as well as for the input of the HHT. This might be caused by the fact that the faults in these laboratory test cases are distinctive. Because the source and target datasets are similar and recorded using the same test setup, a model trained on the source data can deliver an accuracy of 88.4% on the target dataset without the use of transfer learning. By using the intermediate domain, LMMD (91.4%) outperforms the other transfer learning techniques except for MK-MMD, which has a slightly higher accuracy (92.6%). The transfer learning accuracy is better when using the intermediate domain than when using other setups such as HHT and S-transform. Only with a large amount of labeled data, all three techniques can achieve accuracies better than 92%.

However, this setting is far from the real-world scenario in which the model may be required to use a target domain that is completely different from the source dataset.

Table 18: Accuracies of semi-supervised learning. Source and target are different datasets of the Case Western Reserve University – The best results were obtained using only target data and the intermediate domain (93.4%). This is followed by the intermediate domain and MK-MMD (92.6%) and with LMMD (91.4%). Accuracies better than 92% could be achieved for all setups when all target data is used as labeled data.

Setup	Transfer type	Only source domain (%)	TLA1: Labeled data in:			TLA2	Target (%)	All target (%)
			Source (%)	Target (%)	Source and target (%)	Unsupervised source, followed by labeled target (%)		
Intermediate Domain	LMMD	88.4	88.1	**91.4**	90.3	90.0	**93.4**	96.4
	MMD	88.4	88.8	90.8	89.2	90.0	**93.4**	96.4
	Wasserstein	88.4	66.8	79.2	82.4	89.0	**93.4**	96.4
	CORAL	88.4	86.2	90.8	89.0	89.3	**93.4**	96.4
	MK-MMD	88.4	87.4	86.5	88.2	92.6	**93.4**	96.4
HHT	MMD	72.0	68.4	76.5	80.9	70.4	68.1	92.3
	Wasserstein	72.0	66.9	61.2	74.3	67.3	68.1	92.3
	CORAL	72.0	68.6	71.3	74.6	69.2	68.1	92.3
	MK-MMD	72.0	69.4	70.4	72.5	71.2	68.1	92.3
S-transform	MMD	81.4	80.1	74.5	79.1	76.2	73.6	96.6
	Wasserstein	81.4	66.1	71.1	74.3	76.6	73.6	96.6
	CORAL	81.4	76.2	78.6	80.4	76.4	73.6	96.6
	MK-MMD	81.4	81.9	78.7	82.9	76.3	73.6	96.6

3. Comparison of Both Scenarios

In both scenarios, the best transfer learning accuracy is achieved using the intermediate domain and LMMD (unsupervised 87.6% and semi-supervised 91.4%). For the unsupervised case, this is the best achievable accuracy. However, this is not the case for the semi-supervised case. Here, using only the labeled target datasets of the intermediate domain reaches the best results, with an accuracy of 93.4%. The results of the HHT and the S-transform are similar for both scenarios. In addition, the high accuracy of the usage of only a pretrained network with labeled source data indicates, on the one hand, that the intermediate domain is stable enough to be used on these two datasets without modifications. However, on the other hand, it also pointed out that the intermediate domain is capable of doing a transfer learning by itself.

7.2.3.3 Case Study 2: Classification of Different Test Environments

This case study employs two very distinct datasets. The Case Western Reserve University dataset is the target dataset. The source is a dataset of the University of Paderborn. Bearings, rotational speed, sampling frequency, and test environment differ between the two domains.

1. Unsupervised-Only Scenario

It has been found that in this test scenario, the accuracy from using purely unsupervised data with not using DA is only 47.5% with the intermediate domain. The accuracy is further reduced when HHT and S-transform are used. This can be explained by the fact that in this scenario the target data differs in a number of factors from the source data. However, utilizing DA in conjunction with LMMD and CORAL enhances the accuracy to some extent. In this case, CORAL (70.4%) outperforms LMMD (63.2%). Other techniques and combinations do not achieve usable accuracy.

Table 19: Accuracies of only unsupervised learning. Source is a dataset of University of Paderborn dataset and target is a dataset of Case Western Reserve University: Only transfer learning with the intermediate domain and CORAL (70.4%) or LMMD (63.2%) provides good result. The accuracy of any other setting is considerably inferior to that of the CORAL or LMMD settings.

Setup	**Only source domain (%)**	**Domain adaption with a loss function**				
		LMMD (%)	**MMD (%)**	**Wasserstein (%)**	**CORAL (%)**	**MK-MMD (%)**
Intermediate Domain	47.5	63.2	41.1	41.3	**70.4**	33.7
HHT	0.0	-	0.0	33.5	18.5	0.0
S-transform	30.8	-	32.3	35.3	41.0	26.5

2. Semi-Supervised Scenario

Because the target dataset differs from the source, using the pre-trained source network directly does not produce satisfactory results. However, the intermediate domain yields the highest accuracy (50.8%). LMMD surpasses the MMD and Wasserstein approaches in both proposed models (Model 1 and Model 2). The usage of the intermediate domain together with the LMMD is the best performing combination for the case of 80% unlabeled data. It has an accuracy of 89.4%.

It is also discovered that by using all of the target data as labeled data, the classification accuracy can be improved to 96.2% with the help of the intermediate domain. This can be increased when using the S-transform to 98.8%, which is the best accuracy among all combinations of this test scenario. However, when only 20% of the data is labeled, the intermediate domain significantly outperforms the HHT and S-transform.

Table 20: Accuracies of semi-supervised learning. Source is a University of Paderborn dataset and target is a Case Western Reserve University dataset - The transfer learning approach TLA2 with the intermediate domain and LMMD produces the highest results (89.4%). The results of using HHT and S-Transform are always worse than the intermediate domain, except when all data is considered as labeled data. Here the S-transform achieves 98.8%.

Setup	**Transfer type (%)**	**Only source domain (%)**	**TLA1: Labeled data in:**			**TLA2**	**Target (%)**	**All target (%)**
			Source (%)	**Target (%)**	**Source and target (%)**	**Unsupervised source, followed by labeled target (%)**		
Intermediate Domain	LMMD	50.8	74.6	76.2	84.0	**89.4**	88.2	96.2
	MMD	50.8	52.9	62.4	80.8	88.0	88.2	96.2
	Wasserstein	50.8	43.3	71.9	68.7	86.8	88.2	96.2
	CORAL	50.8	64.4	75.7	84.2	87.1	88.2	96.2
	MK-MMD	50.8	53.0	80.5	80.5	69.7	88.2	96.2
HHT	MMD	0.0	0.0	52.3	41.2	71.5	75.0	89.9
	Wasserstein	0.0	17.2	55.6	50.8	64.3	75.0	89.9
	CORAL	0.0	23.5	54.2	51.5	69.1	75.0	89.9
	MK-MMD	0.0	0.6	54.2	44.5	72.0	75.0	89.9
S-transform	MMD	35.2	15.3	66.0	61.3	73.3	63.8	98.8
	Wasserstein	35.2	33.2	59.5	63.0	66.3	63.8	98.8
	CORAL	35.2	33.8	75.0	68.6	73.6	63.8	98.8
	MK-MMD	35.2	19.8	43.9	65.9	74.1	63.8	98.8

3. Comparison of Both Scenarios

In both scenarios, the best accuracy can be reached with transfer learning and the intermediate domain as input (70.4% for unsupervised, and 89.4% for semi-supervised). For the unsupervised scenario, CORAL as transfer loss function has the best results (70.4%), followed by LMMD (63.2%). In the semi-supervised scenario, LMMD is again the best transfer loss function, with an accuracy of 89.4%. Using images created with the HTT as input leads to bad results in both scenarios. Based on the best results of this case study when using the intermediate domain, it is also shown that the intermediate domain can be used with different bearing types without a reparameterization. In addition, its ability to perform transfer learning is shown by the slightly improved accuracies compared to other techniques when not making a domain adaptation.

7.2.3.4 Case Study 3: Classification with a Real-World Example

The source domain of this real-world test scenario is the dataset of the University of Paderborn's test laboratory, which contains both simulated and genuine bearing faults. The target domain data is obtained from a grinding spindle, where all faults are genuine. The datasets differ in terms of sample frequency, bearings, test environment (real-world vs. lab), and rotational speed. The rotational speed plays an important role because the target dataset was recorded with three rotational speeds that varied up to 90%.

1. Unsupervised-Only Scenario

The intermediate domain achieved the highest accuracy. Thereby, CORAL performs best (66.2%), followed by MK-MMD (55.2%) and LMMD (53.6%). Furthermore, using the S-transform as an input can produce the highest accuracy without doing a domain adaptation. Nevertheless, the accuracy of all combinations is too poor to be used in real-world applications.

Table 21: Accuracies of only unsupervised learning. Source is a University of Paderborn dataset and target is a spindle dataset with different rotational speeds. The intermediate domain in conjunction with CORAL yields the best accuracy (66.2%). This is followed by LMMD (53.6%). The S-transform produces the most accurate results without transfer learning, but the accuracy is still insufficient.

Setup	**Only source domain (%)**	**Domain adaption with a loss function**				
		LMMD (%)	**MMD (%)**	**Wasserstein (%)**	**CORAL (%)**	**MK-MMD (%)**
Intermediate Domain	38.3	53.6	48.5	35.1	**66.2**	55.2
HHT	36.2	-	33.6	50.4	35.3	33.0
S-transform	58.0	-	45.1	32.6	55.7	43.8

2. Semi-Supervised Scenario

Table 22: Accuracies of semi-supervised learning. Source is a University of Paderborn dataset and target is a spindle dataset with different rotational speeds. The intermediate domain and LMMD in conjunction with transfer learning (TLA1 - Target) provide the best results (87.7%). With the intermediate domain input, other loss functions also provide encouraging results for Model 2, but the results are worse than the ones of LMMD. The intermediate domain consistently outperforms the S-transform, which in turn outperforms the HHT.

Setup	**Transfer type**	**Only source domain (%)**	**TLA1: Labeled data in:**			**TLA2**	**Target (%)**	**All target (%)**
			Source (%)	**Target (%)**	**Source and target (%)**	**Unsupervised source, followed by labeled target (%)**		
Intermediate Domain	LMMD	37.8	61.4	**87.7**	85.8	83.9	84.0	87.7
	MMD	37.8	52.2	58.7	73.2	81.0	84.0	87.7
	Wasserstein	37.8	37.2	77.9	57.5	87.2	84.0	87.7
	CORAL	37.8	63.5	76.5	85.2	86.5	84.0	87.7
	MK-MMD	37.8	52.3	59.0	75.6	82.4	84.0	87.7
HHT	MMD	35.4	36.0	52.4	56.2	65.0	51.7	71.4
	Wasserstein	35.4	38.6	47.5	39.2	49.7	51.7	71.4
	CORAL	35.4	41.0	56.6	58.1	57.7	51.7	71.4
	MK-MMD	35.4	35.4	48.9	48.4	55.5	51.7	71.4
S-transform	MMD	59.4	53.3	56.1	58.6	60.6	51.2	70.2
	Wasserstein	59.4	38.9	67.6	52.7	61.1	51.2	70.2
	CORAL	59.4	59.1	54.0	65.1	59.4	51.2	70.2
	MK-MMD	59.4	46.9	56.8	51.7	57.8	51.2	70.2

In the semi-supervised test scenario, the highest accuracies are reached using the intermediate domain and LMMD in combination with a training process according to TLA1 and by only using additional labeled target measurements. Here, an accuracy of 87.7% can be reached. This is a 20.1% higher accuracy than the best accuracy without the intermediate domain (S-transform with Wasserstein loss: 67.6%). The confusion matrix also reveals that the 87.7% accuracy is mainly because some inner ring faults are detected as outer ring faults. These two circumstances confirm that this approach can be applied in real-world applications. This test scenario also demonstrates that by using the intermediate domain, an accuracy of 84.0% can be attained by using only a small, labeled dataset and with no transfer learning. This represents an improvement of 32.3%. A significant number of samples are required to employ HHT and the S-transform. This is demonstrated by an accuracy of around 50% on a subset of all samples and an improvement to an accuracy around 70% when using all samples as labeled data. However, this accuracy is still lower than the accuracies of the intermediate domain.

3. Comparison of Both Scenarios

As in the two previous case studies, the intermediate domain reaches the highest accuracy in both scenarios (66.2% unsupervised and 87.7% semi-supervised). HHT and S-transform have a much lower accuracy (50.4% unsupervised and 67.6% semi-supervised). In addition, these accuracies are too low for a real-world implementation. This demonstrates again that the intermediate domain can be used on different bearing types without reparameterization. Among the two setups, the S-transform outperforms the HHT in both scenarios. Using LMMD and CORAL as transfer functions result in the best results. However, of all the use cases in this case study, the only one with a suitable accuracy for real-life use with small datasets is the semi-supervised use case, which uses an intermediate domain and LMMD.

7.2.4 Conclusion

The evaluation, which has been performed with the help of three case studies, shows that the presented intermediate domain and the LMMD are effective for the execution of transfer learning tasks. In particular, the hybrid approach based on the intermediate domain leads to strong improvements compared to pure data-driven approaches like HTT and S-transform. When only a small subset of the target domain data is used, the intermediate domain results are the most accurate in all three test cases – for supervised learning as well as for transfer learning. Based on these three different use cases, it is verified that the intermediate domain is stable enough to be used for different bearing types without any reparameterization. In addition, when using models trained with the source domain dataset only, the results are also the best in most cases, indicating its supportive influence on transfer learning. This verifies the effectiveness of the intermediate domain and is consequently an answer for RQ3, which asks for the characteristics of an intermediate domain that is stable enough to be used on different bearing types without changing the intermediate domain parametrization or making big changes to a subsequent machine learning model.

Neither of the presented training models for the semi-supervised scenario is superior to the other. Compared to the traditional MMD and Wasserstein methods, the accuracy can be improved by using the LMMD loss function. In addition, LMMD outperforms CORAL and MK-MMD in most of the cases. An exception are two unsupervised learning cases where CORAL outperforms LMMD. Nevertheless, even by using these improved methods, pure unsupervised training is not always feasible. In case study 1, where target and source domains are comparable, it may be a choice, but the results are insufficient for use in real-world scenarios. The case studies also show that transfer learning may not always produce optimal results. Sometimes a negative transfer occurs, and re-training with only a small amount of target data can lead to more accurate results. As described in Section 2.4.2, this can be traced back to two facts. First, it might happen when samples from the target domain are similar and distinctive, as in the case of artificial defects in the Case Western Reserve University dataset, where, for instance, all faults have the same size. Second, at least the first convolutional layers are often fixed in the case of transfer learning. It may be required to retrain more than the last layers if the source and target domains are highly dissimilar. However, in the given use cases, this is not the case.

Case Study 2 demonstrates that using the intermediate domain and LMMD together improves accuracy by around 15% in comparison of using conventional semi-supervised learning techniques. However, in the real-world scenario (case study 3), where many different aspects, such as various rotational speeds and noise, are present in the target domain, it is possible to see the true potential of the suggested approach. The intermediate domain in conjunction with LMMD yields the most accurate results (accuracy of 87.7%). Considering these aspects, it can be contemplated that the results are good enough for the usage in real-world predictive maintenance applications, especially when the target domain is based on datasets with different rotational speeds.

To summarize, this exploration also verifies the classification approach of Chapter 5, which is the answer for RQ1 that asks for a new classification method, which can take benefits of a dataset of a different bearing type for a partly labeled target dataset that is collected under different process conditions.

7.3 Benchmark: Transfer Learning for Bearing Fault Classification

7.3.1 Introduction

This section benchmarks the domain adaptation classification framework presented in Chapter 5 against the research work of Cheng et al. [83]. It is important to mention that the presented approach is intended to be a general approach for different bearing types. Therefore, the classification framework has not been adapted in any way. This is true for the parameters of the intermediate domain and the CNN.

As a first step, this section explains the research of Cheng et al. and its setup in detail. This is followed by a comparison of the results to the ones of the approach presented here. Finally, a conclusion is presented based on the results.

7.3.2 Benchmark Description

The research of Cheng et al. [83] uses the datasets of the CWRU for pure data-driven unsupervised transfer learning. The authors investigated different transfer learning scenarios with these datasets. One scenario is similar to the case study presented in Section 7.2.3.2, where the domain adaption is applied between the different but similar bearings of the fan-end and the drive-end side. In addition, both datasets contain samples of the same process conditions. As described in Section 3.3.2.5, there is no other research that is appropriate for a direct benchmark. The research of Cheng et al. was chosen as a benchmark because, to the best of the author's knowledge, this is the only transfer learning approach between different bearing types that uses the publicly available CWUR datasets. This makes it possible to make a direct benchmark.

The transfer learning of their research work has the following setup:

- Two transfer learning tasks: The first is to transfer knowledge from the bearings of the drive-end side to the fan-end side, and the second is to transfer from the fan-end side to the drive-end side.
- Each transfer task has four health conditions: inner ring fault, outer ring fault, ball fault, and normal.
- The exact split between training and test samples is unknown.
- In addition, it is also unknown if the split is bearing instance-based or sample-based. This means that it is unknown whether samples of a particular bearing instance are included in the test, training, or both datasets.
- Five different test scenarios were run for 5,000 iterations each. The best accuracy for each scenario has been chosen. Based on the resulting five accuracies, the average and the 95% confidential interval of the classification accuracy were calculated to rate a specific transfer learning approach.

This setup is used for six different transfer learning approaches. Some are classical machine learning approaches based on an SVM and others are deep learning approaches based on a CNN. Unfortunately, their research does not present all used parameters, and the authors did not respond to contact requests. However, the missing parameters are not important for establishing a direct benchmark based on the resulting accuracies between the proposed approach and the work of Cheng et al. A detailed description of the authors' approaches is given below.

7.3.2.1 SVM

Cheng et al. used three different transfer learning approaches based on an SVM. The detailed setup of these approaches is unknown. The transfer functions are specified as follows:

- Transfer component analysis (TCA): This approach uses TCA to bring the source and the target domain close together. The TCA algorithm is based on an extended version of MMD.
- JDA: As described in Section 3.3.2.4 in brief, JDA is also an extension of the MMD. It is a sum of the classical MMD and an MMD, which is calculated with the conditional distribution of each category as an input.
- CORAL: This method is described in detail in Section 2.4.6.5.

7.3.2.2 CNN

The research of Cheng et al. is not only focused on traditional machine learning but also on deep learning techniques. They presented two transfer learning approaches that are based on a CNN. The CNN uses the raw vibration data as input. For this reason, 1D convolutional and pooling layers are used. The detailed architecture with all used layers and their parameters of the CNN is shown in Table 23.

Table 23: This table shows the detailed architecture with all used layers and their parameters of the CNN used by Cheng et al. Each layer of the CNN is listed with its type: the output shape of each layer (none has to be replaced with the number of images that are used in a batch and is therefore dependent on the training parameter batch size) and the used activation functions. Unfortunately, the exact size of the input and of the layers is not mentioned in the paper.

Layer	Type	Output Shape	Activation
0	InputLayer	(None, [Unknown], 1)	
1	Conv1D	(None, [Unknown], 8)	ReLu
2	MaxPooling1D	(None, [Unknown], 8)	
3	Conv1D	(None, [Unknown], 16)	ReLu
4	MaxPooling1D	(None, [Unknown], 16)	
5	Dense (FC1)	(None, 128)	ReLu
6	Dense (FC2)	(None, 4)	Softmax

In their research, Cheng et al. used the following three CNN-based approaches to classify the bearings in the target domain:

- CNN: Here, no transfer learning is used. The network is pre-trained with the source domain and is directly tested with the target domain data.
- Domain adaption network (DAN): This method is similar to the one presented in Section 5.4 and used in Section 7.2.3, wherein the CNN is pre-trained with the source dataset. Afterward, the weights and biases of the convolutional layers are frozen, and only the dense layers are trained during transfer learning. Here, they used MK-MMD on the output of the first dense layer (Layer 5) to adapt the two domains.

- Wasserstein Distance based Deep Transfer Learning (WD-DTL): This is the novelty presented in the paper of Cheng et al. The difference between the DAN and the WD-DTL is that it uses the Wasserstein distance to accomplish the transfer learning between the source and target domain.

7.3.3 Benchmark

In order to compare the classification approach presented in this thesis with the work presented above, a setup with the following specifications for the unknown parameters has been used:

- The dataset has been split into 70% training data and 30% test data, as in the case studies in Section 7.2, since the split ratio from [83] is not known.
- The assignment of the different bearings to the training and test dataset is also not known. Therefore, five random assignments into training and test data have been chosen. The average of the results as well as the 95% confidential interval of the classification accuracy have been calculated.

This benchmark has been done for the fan-end side as the source domain and the drive-end side as the target domain and vice versa. The results of both transfer learning tasks are listed in Table 24.

Table 24: The resulting accuracies of the transfer learning tasks from drive-end bearing data to fan-end bearing data and vice versa. For each direction, five different assemblies of the data were used. In addition, the average and the 95% confidential interval of the five runs are listed. The runs are ordered according to the LMMD accuracy.

Transfer Task	Run	Intermediate domain direct accuracy (%)	LMMD accuracy (%)	Average intermediate domain direct (%)	Average LMMD (%)
Drive-End -> Fan-End	1	60.66	73.82	58.22 (± 2.51)	61.37 (± 6.55)
	2	60.69	62.78		
	3	59.47	58.03		
	4	55.39	56.18		
	5	54.87	56.05		
Fan-End -> Drive-End	1	69.94	78.70	67.76 (± 3.19)	72.34 (± 3.96)
	2	70.44	73.43		
	3	65.78	72.31		
	4	70.37	71.09		
	5	62.25	66.15		

The accuracy of the transfer task from fan-end to drive-end is about 10% better than in the opposite direction. When considering the confusion matrix (see Appendix A.4), it can be seen that for all transfers from drive-end to fan-end, the classification accuracy for bearings in a normal condition is bad. In one case, no sample was classified correctly (see Figure 54a). By contrast, the accuracy of the normal condition for the transfer from the fan-end to the drive-end side is much better. In four out of five runs, all bearings in normal condition were classified correctly (see also Appendix A.4). Figure 54b shows an example where every bearing in the normal condition was classified correctly.

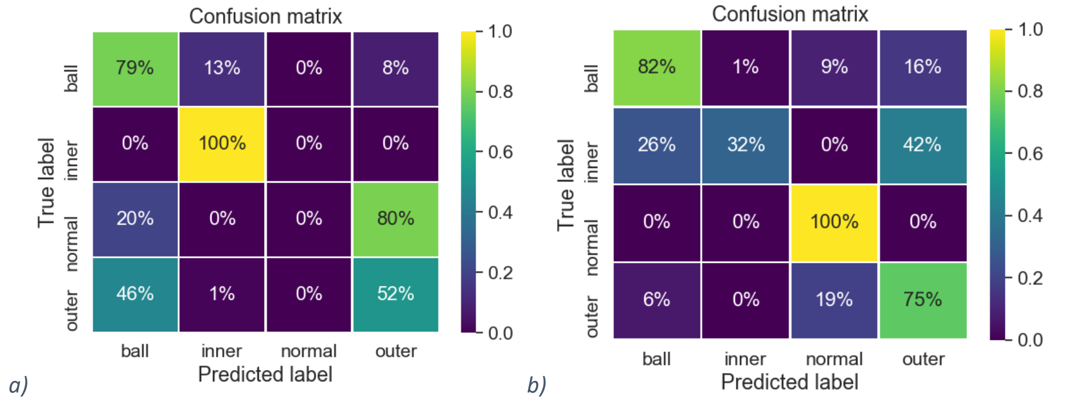

Figure 54: Examples showing the confusion matrices of the third run for both transfer learning directions. The third run is only used to show a confusion matrix of a random result. All other confusion matrices are illustrated in Appendix A.4. a) shows the confusion matrix for drive-end to fan-end with an accuracy of 58.03% and b) shows the confusion matrix for fan-end to drive-end with a total accuracy of 72.13%.

Table 25 presents a comparison between the results of the two transfer tasks carried out in this thesis and the research of Cheng et al. The displayed accuracies clearly indicate that for the case of the pure data-driven CNN model without transfer learning, the use of the CNN model, developed in Section 5.4.2, together with the intermediate domain of Chapter 4, is superior (average accuracy of 62.99%) to the model of the research paper (average accuracy of 39.51%). By combining this approach with the LMMD loss function for transfer learning, an even higher accuracy is reached. For the transfer learning from the fan-end side to the drive-end side, the best accuracy (72.34%) is reached. The second-best accuracy is reached for the transfer learning from the drive-end side to the fan-end side. LMMD achieves the best average accuracy (66.85%) for both transfer directions.

Table 25: Results of transfer learning with an intermediate domain and LMMD compared to the results of Cheng et al. The overall best result is achieved with the LMMD transfer function and the proposed intermediate domain, with an accuracy of 66.85%.

Author	**Transfer Approach**	**Drive-End -> Fan-End**	**Fan-End -> Drive-End**	**Average**
Cheng et al. [83]	TCA	19.05	20.45	19.75
	JDA	57.35 (± 0.47)	66.34 (± 4.47)	61.85 (± 2.47)
	CORAL	47.97	39.87	43.92
	CNN	39.07 (± 2.22)	39.95 (± 3.84)	39.51 (± 3.03)
	DAN	56.89 (± 2.73)	55.97 (± 3.17)	56.43 (± 2.95)
	WD-DTL	64.17 (± 7.16)	64.24 (± 3.87)	64.20 (± 5.52)
This thesis	Intermediate domain	58.22 (± 2.51)	67.76 (± 3.19)	62.99 (± 2.85)
	LMMD	61.37 (± 6.55)	72.34 (± 3.96)	**66.85 (± 5.26)**

7.3.4 Conclusion

The best average accuracy of 66.85% for the transfer task from fan-end -> drive-end and vice versa is achieved with the proposed intermediate domain combined with transfer learning by means of LMMD. This is better than the best accuracies of Cheng et al. who achieved an accuracy of 64.20% for the same task. It is also remarkable that when using the intermediate domain directly without any transfer learning an average accuracy of 62.99% is reached. This accuracy can be compared to the accuracy of 39.51% of the CNN in Cheng et al. since neither setup utilizes transfer learning. In addition, for the task of knowledge transfer from fan-end to drive-end, the intermediate domain without any used target data for training has even better accuracies than all approaches in the paper by Cheng et al. To summarize, the intermediate domain, as well as transfer learning with the loss function LMMD, are superior to the techniques presented by Cheng et al. It should be mentioned that the benchmark used datasets in the source and target domain with the same process conditions and similar bearing types. This leads to the fact that the selected approach cannot show its full strength, which could lead to a higher accuracy difference, as can be seen in Sections 7.2.3.3 and 7.2.3.4, where different process conditions and bearing types are used in each domain. In addition, no optimizations of the classification framework for the given use case have been carried out.

Unfortunately, it has been shown that when LMMD is used for transfer learning, the transfer task from drive-end to fan-end does not classify normal bearings correctly. Since neural networks behave like "black boxes" [192] and Cheng et al. did not publish any confusion matrices and also did not release them upon request, it is unknown whether this is a problem based on the dataset or the technology used.

To summarize, this benchmark has shown that the transfer learning approach presented in Chapter 5 delivers better accuracies than the current research of Cheng et al. This benchmark was achieved under the same conditions, so the results are directly comparable. In addition to the exploration of chapter 7.2, this benchmark also answers RQ1, which requires a solution for the classification of the bearing health of different bearing types based on partly labeled target datasets.

7.4 Benchmark: Transfer Learning for Remaining Useful Life of Bearings

7.4.1 Introduction

This section presents a benchmark based on the IEEE PHM 2012 data challenge for the transfer learning-based RUL approach of Chapter 6. This challenge took place in 2012 and was hosted by the IEEE Reliability Society and the FEMTO-ST Institute. The focus was on the estimation of the RUL of bearings. Participation was open to both professional (industry) and scientific (university) attendees [178]. To the author's knowledge, there is currently no better or more widely used benchmark and reference dataset available. However, the only two recent transfer learning approaches targeting different bearing types by Xia et al. [158] and Huang et al. [159] do not use this benchmark. Each of them uses its own benchmark, which is not explained in detail. Therefore, it is impossible to benchmark the presented RUL approach against their approaches.

This section is divided into benchmark description, benchmark execution, detailed benchmark analysis, and a conclusion. The detailed benchmark analysis examines the influence of the constraint described in Section 6.3 on particular datasets. This constraint is that the intermediate domain is only capable of analyzing bearings that are at least at degradation stage 3. The usage of this detailed benchmark analysis is in contrast to the structure of the two case studies for the classification task (Section 7.2 and Section 7.3). They do not need this section because the classification approach does not have any constraints.

Large parts of Section 7.4.3 and 7.4.4 of this benchmark have already been published in a research article by the author [112].

7.4.2 Benchmark Description

This benchmark was designed to estimate the RUL of bearings. For this purpose, the datasets of the FEMTO-ST institute, which was already presented in Section 3.3.3.2, was used. Because these datasets are based on accelerators with a maximum sampling frequency of 25.6 kHz, it may be possible that other approaches that use the entire available frequency spectrum provide better results than the proposed approach, which is optimized for frequencies up to 5 kHz. According to the given benchmark setup of the challenge, the datasets were assigned into training and test data, as shown in Table 26. In addition, a different number of measurements were removed from the end of the test datasets so that each test dataset had a different RUL. Finally, the solutions of the different attendees of the challenge were evaluated according to the PHM score in Appendix A.5.1.

Table 26: Assignment of the different datasets to test and training data.

Datasets	**Operating Conditions**		
	1,800 rpm; 4,000 N load	**1,650 rpm; 4,200 N load**	**1,500 rpm; 5,000 N load**
Learning set	Bearing1_1	Bearing2_1	Bearing3_1
	Bearing1_2	Bearing2_2	Bearing3_2
Test set	Bearing1_3	Bearing2_3	Bearing3_3
	Bearing1_4	Bearing2_4	
	Bearing1_5	Bearing2_5	
	Bearing1_6	Bearing2_6	
	Bearing1_7	Bearing2_7	

7.4.3 Benchmark Execution

This benchmark was run under the identical settings as in the IEEE PHM 2012 Data Challenge. Additionally, as described in Section 6.4.4., the RUL network was pre-trained with the drive-end dataset of Case Western Reserve University (see Section 3.3.2.2). The pre-trained network was then trained using these parameters:

- Data usage: The input data was formatted according to the settings proposed in Chapter 6, which include a total input size of 85 measurements (*n = 85*) where every second measurement (*s = 2*) was skipped during the input preparation.
- Intermediate domain: The rotational speed combined with the used bearing results in the following characteristic fault frequencies: Cage fault: 13 Hz, ball fault: 108 Hz, outer ring fault: 168 Hz, and inner ring fault: 222 Hz. The intermediate domain uses four harmonics, which results in a total maximum used frequency of 888 Hz for the inner ring fault. As defined in Section 6.1, one requirement is to have a solution that can be used in use cases with current industrial triaxial accelerometers, which have a sampling rate of about 5,000 Hz. This requirement is fulfilled by having a maximum used fault frequency of 888 Hz.
- Training settings:
    - Batch size: For the training, a batch size of 120 was employed. Due to the limits of the hardware being used, a bigger batch size could not be employed.
    - Learning rate: A learning rate of 0.0005 was used, as proposed in the literature [133].
    - Optimizer: During training, an Adam optimizer with the mean squared error (MSE) loss function was employed. This is based on the recommendation of Liu et al. [179] that MSE is the most sensitive loss function for measurement errors among all common loss functions.
    - Training epochs: 300 epochs were used for the training, since no better results are reached in the result of the loss function afterward.

The results of this evaluation are presented using the relative error (Er), its mean, and the PHM scoring algorithm (see Appendix A.5.1) in Table 27. In addition, the results of Sturisno et al. [155] (academic challenge winner), Porotsky and Bluvband [193] (industrial challenge winner), Zheng [194] (current research), and Zhang et al. [157] are also presented. The approach of Zhang et al. is currently the best in terms of mean relative error and PHM score. In addition, they compared their approach to other recent approaches, which had PHM scores from 0.26 to 0.62. A recent approach by Xu et al. [195] achieved a score of 0.84. However, since it is not peer-reviewed, it is not considered in this thesis. Each of the aforementioned research used purely data-driven approaches without the usage of physical parameters.

Table 27: The relative error (Er), its mean, and the score of the different RUL approaches is shown in this table. Sturisno et al. and Porotsky and Bluvband are the winners of the IEEE PHM 2012 Data Challenge. They have scores of 0.3066 and 0.28. An example for a current approach is Zheng with a score of 0.2992. The best current approach by Zhang et al. has a score of 0.64. The proposed RUL approach has a score of 0.35, which represents a value approximately in the average. However, the approach is the worst when considering the mean of the Er due to two outliers (bearing 1_6 and bearing 2_5). If these two bearings were ignored, the presented approach would be good in this area as well.

Bearing	**Sutrisno et al. (%)**	**Porotsky and Bluvband (%)**	**Zheng (%)**	**Zhang et al. (%)**	**Proposed RUL Framework (%)**	**Proposed RUL Framework without 1_6 and 2_5 (%)**
Bearing 1_3	97	N/A	92.44	2.27	29.27	29.27
Bearing 1_4	80	N/A	100	5.6	-78.35	-78.35
Bearing 1_5	9	N/A	20.43	12.42	-159.24	-159.24
Bearing 1_6	-5	N/A	7.76	10.96	-6413.71	N/A
Bearing 1_7	-2	N/A	82.29	-22.46	35.37	35.37
Bearing 2_3	64	N/A	82.93	0.99	-0.7	-0.7
Bearing 2_4	10	N/A	3.22	5.76	-124.18	-124.18
Bearing 2_5	-440	N/A	58.77	25.89	-919.58	N/A
Bearing 2_6	49	N/A	5.63	-10.85	8.16	8.16
Bearing 2_7	-317	N/A	-121.94	1.72	12.13	12.13
Bearing 3_3	90	N/A	-54.38	-3.66	-0.96	-0.96
Mean	105.73	N/A	57.25	9.32	707.42	40.76
Score	0.3066	0.28	0.2992	0.64	0.35	0.43

The PHM score is the used metric for the IEEE PHM 2012 Data Challenge. The proposed approach surpasses the two competition winners and many other approaches, such as Zheng [194], when using this metric. However, when using the relative error, the results of the suggested approach are the worst. As shown in Table 28, the mean relative error of 707.42% results from two outliers with a relative error of -6413.71% (bearing 1_6) and -919.58% (bearing 2_5). Without the two outliers, the benchmark results in a mean relative error of 40.76%. This is again a good result. It is conspicuous that Sturisno et al. , also have the highest relative error for bearing dataset 2_5.

The two outliers of the proposed approach (bearing 1_6 and bearing 2_5) have a negative *Er*. A negative *Er* means a too large, estimated RUL (see Eq. (29)). A possible reason for this could be that the behavior of those two test datasets is not covered by the trained network, which can be the result of the only few training datasets. This could lead to a trained network that is not general enough. Therefore, the degradation characteristics of these two datasets is not covered by the trained network. The complete datasets of these two bearings were added to the training data to validate this. It turned out that this did not result in any improvements, as would have been expected. Another possibility for this could be the limitation of the used approach mentioned in Section 6.3, where the bearing has to be at least at degradation stage 3. This assumption will be examined in the following subchapter.

Table 28: Detailed summary of the results for the proposed RUL framework. Here, the nominal as well as the estimated RUL are shown. These two values are used to calculate the relative error Er. Er is then used to calculate A. The sum of all As is the calculated PHM score. For a detailed explanation of the calculation, see Appendix A.5.1.

Bearing	Nominal RUL (s)	Estimated RUL (s)	Er (%)	A
Bearing 1_3	5730.0	4052.6	29.27	0.36
Bearing 1_4	2890.0	5154.4	-78.35	0.00
Bearing 1_5	1610.0	4173.8	-159.24	0.00
Bearing 1_6	1460.0	95100.2	-6413.71	0.00
Bearing 1_7	7570.0	4892.6	35.37	0.29
Bearing 2_3	7530.0	7582.6	-0.70	0.91
Bearing 2_4	1390.0	3116.2	-124.18	0.00
Bearing 2_5	3090.0	31504.9	-919.58	0.00
Bearing 2_6	1290.0	1184.7	8.16	0.75
Bearing 2_7	580.0	509.6	12.13	0.66
Bearing 3_3	820.0	827.8	-0.96	0.88
			Mean: 707.42	PHM score: 0.35

Except for the aforementioned outliers, there is a good matching between nominal and estimated RUL for all other bearings. The relative error of bearing 2_3 and 3_3 is even less than 1%. The excellent test result of the two test cases cannot be due to similarity since bearing 2_3 has a very high nominal RUL (7530.0 s) and bearing 3_3 has a very low RUL (820.0 s). However, most other test cases also have good results, with a relative error in the lower two-digit range.

There are also recent approaches, e.g., Zhang et al. [157], that have superior results to the one presented. To the best of the authors' knowledge, the input of all superior approaches are features of the time-frequency domain. Unlike the suggested approach, that uses frequencies of less than 1,000 Hz, they utilize the entire frequency range of the datasets as input. This frequency range extends to 12,800 Hz.

Another outcome of this benchmark is the influence of different process conditions. As described in Section 3.3.3.2, the datasets have slightly different process conditions that vary from 1800 rpm combined with a load of 4000 N for the datasets bearing 1_*n* to 1500 rpm combined with a load of 5000 N for bearing 3_*n*. Since the results are very similar regardless of the process conditions the current sample belongs to, it can be assumed that the affiliations are not relevant. This is true for the proposed approach as well as for the other approaches from the literature. In addition, the two outlier datasets also belong to different process conditions.

7.4.4 Detailed Benchmark Analyses

7.4.4.1 Introduction

The benchmark execution revealed that the estimated RUL of most test datasets is more accurate than that of other current solutions. Unfortunately, the estimated RUL is wrong for two datasets (bearing 1_6 and bearing 2_5). This chapter analyzes these datasets in detail. For this purpose, first, an analysis of a reference dataset (bearing 1_4) is given. Afterward, the datasets of bearing 1_6 and bearing 2_5 are analyzed. Finally, a conclusion summarizes the findings.

7.4.4.2 Analysis of Bearing 1_4

This dataset was chosen to study the expected behavior during degradation. The total length of this dataset is 14,280 seconds, and a lifetime of 11,390 seconds was defined as the test position. The dataset is examined in the time and time-frequency domain to obtain an overall picture of the degradation process.

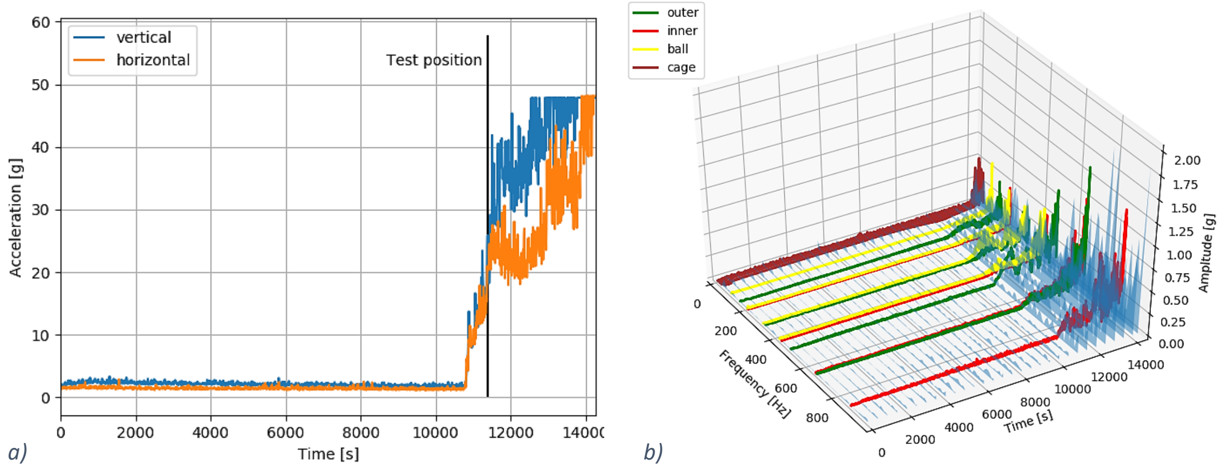

Figure 55: The sensor values of bearing dataset 1_4 over time: Figure a) shows the maximum acceleration value of the horizontal and vertical accelerators of each measurement record. In addition, the test position is marked. Figure b) shows the values of the horizontal sensor in the time-frequency domain. For this purpose, an FFT was performed for the measured values every 500 seconds. Both figures show an increased amplitude of the fault frequencies at the end of the lifetime [112].

Figure 55 shows the measured acceleration values over time. The time domain plot in a) shows the maximum measured value per record. Both accelerometers show increased accelerations at the test position. In addition, the accelerations reveal a rising tendency until the end of the lifetime. These characteristics match those in the time-frequency domain as well, which is shown in b). Figure 55 indicates that the bearing is already in degradation stage 3 at the test position. This can be seen from the already-increased acceleration value as well as through the amplitude of the characteristic fault frequencies in the time-frequency domain. Therefore, the presented intermediate domain-based approach is well-suited to estimating the RUL for this dataset.

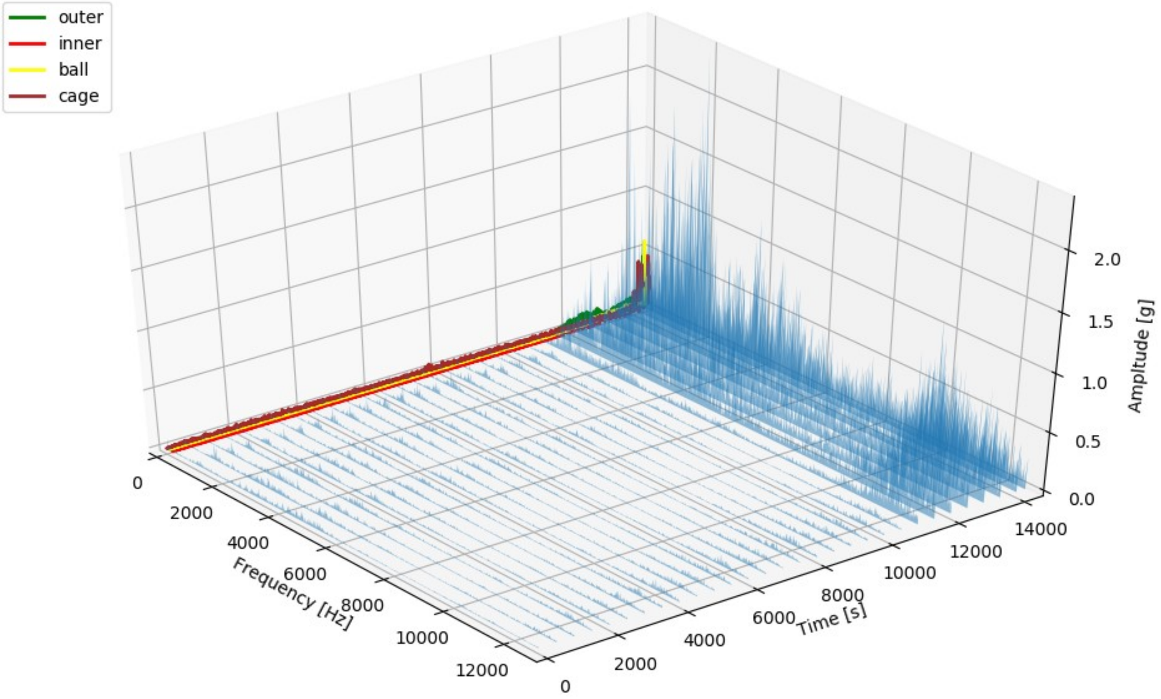

Figure 56: The amplitude of the entire frequency range (up to 12,800 Hz) of the horizontal accelerometer of bearing dataset 1_4 over time. Additionally, to the fault frequencies, amplitudes of other frequencies also rise at the end of the bearing life.

The 3D plot presented in Figure 56 shows that at the end of the bearing lifetime, not only the characteristic fault frequencies and its harmonics have an increased amplitude. Even before the fault frequencies appear, there are amplitudes with an increasing trend on other frequencies. These are the frequency areas from 1000 Hz to 6000 Hz and from 10,000 Hz to 12,000 Hz.

7.4.4.3 Analysis of Bearing 2_5

The benchmark result of bearing 2_5 is a relative error of *Er* = -919.58 % at the test position, which is after a lifetime of 20,020 seconds.

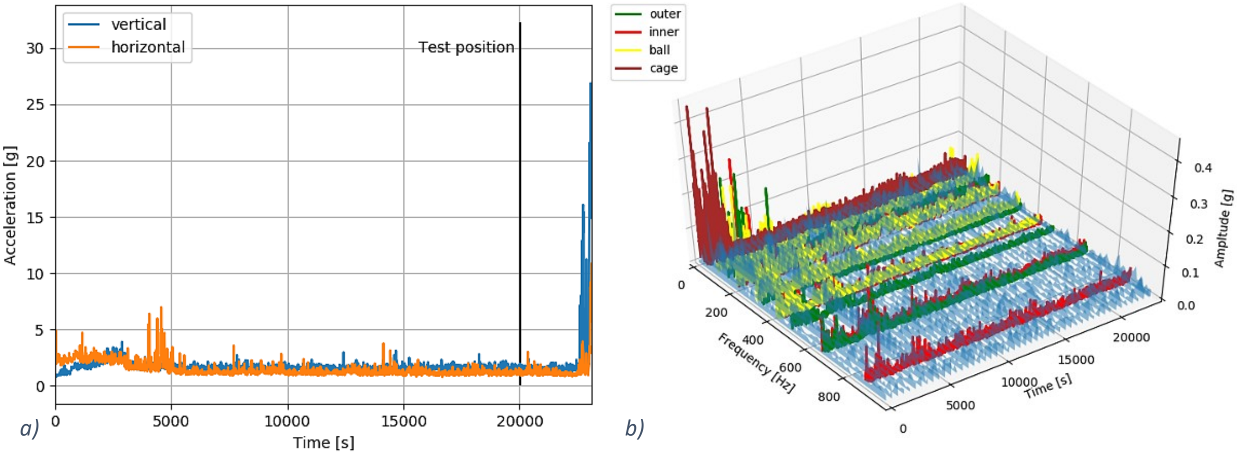

Figure 57: The sensor values of bearing dataset 2_5 over time: Figure a) shows the maximum acceleration value of the horizontal and vertical accelerators of each measurement record. In addition, the test position is marked. Figure b) shows the values of the horizontal sensor in the time-frequency domain. For this purpose, an FFT was performed for the measured values every 500 seconds. A) and b) do not show a special characteristic at the test position, which is after 20,020 seconds. Figure a) shows an increased measured value at the end of the lifetime. Figure b) also shows slightly increased amplitudes towards the end [112].

As shown in Figure 57, this dataset does not have any signs of a degradation of stage 3 at the test position. Only towards the end of the RUL an expected increase of the acceleration values can be seen for both sensors in the time domain. However, only moderate increased amplitudes are recognizable at the characteristic frequencies and their harmonics. This can be a result of a distributed fault. As explained in Section 3.2.4, a distributed fault is a fault that affects the entire bearing and can, for instance, occur due to a lack of lubrication. Such an error does not necessarily have largely increased fault frequencies.

The plot of the entire frequency range in Figure 58 is consistent with Figure 57. There are also only moderate changes in the amplitudes below 1,000 Hz. Significant changes in the amplitude are only visible at higher frequency ranges such as 1,000 Hz, 6,000 Hz, and 12,000 Hz. At the test position itself, the values of the frequency range around 1,000 Hz are just starting to increase. As already seen for bearing 1_4, these frequency ranges have an increasing trend towards the end of the bearing lifetime, which leads to the assumption that these are natural frequencies of the bearing components, as described in Section 3.2.4.

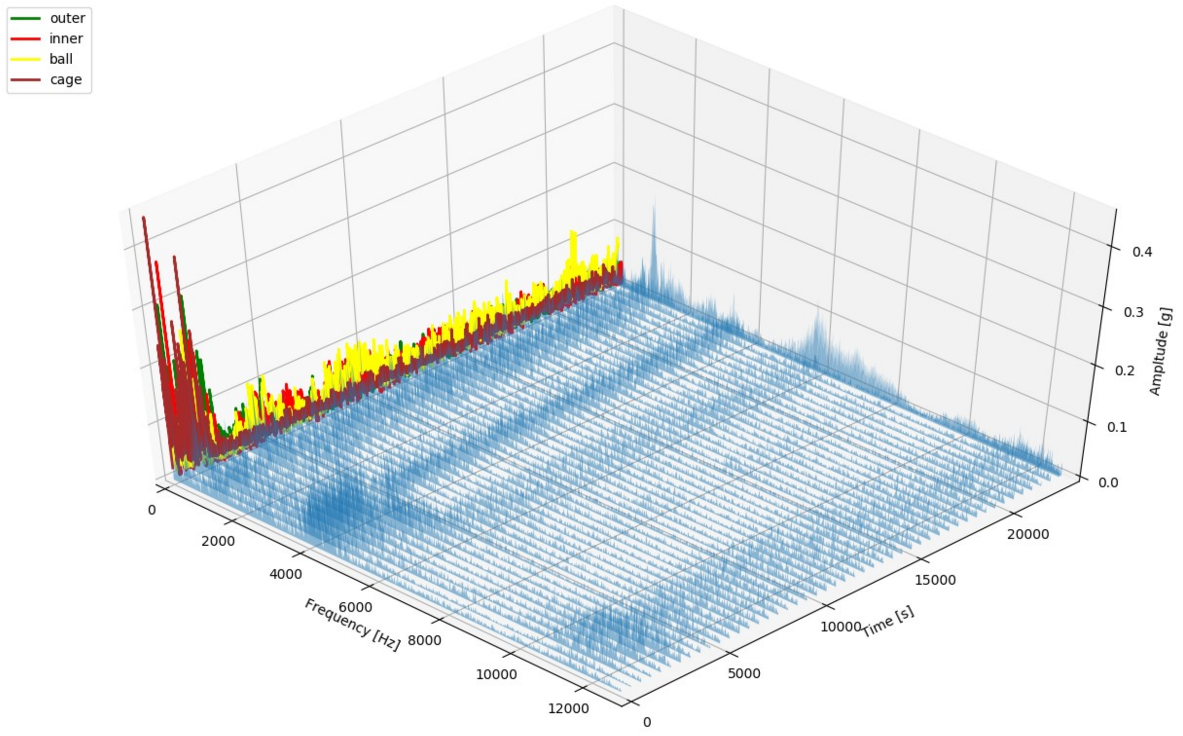

Figure 58: This plot shows the amplitude of the entire frequency range (up to 12,800 Hz) of the horizontal accelerometer of bearing dataset 2_5 over time. There is hardly any increased amplitude at frequencies lower than 1,000 Hz. This is the area of the characteristic fault frequencies. However, the frequency ranges of 1,000 Hz, 6,000 Hz, and 12,000 Hz have increased amplitudes at the end of the RUL.

7.4.4.4 Analysis of Bearing 1_6

The benchmark result of bearing 1_6 is a relative error of *Er* = -6413.71 % at the test position, which is after a lifetime of 23,020 seconds.

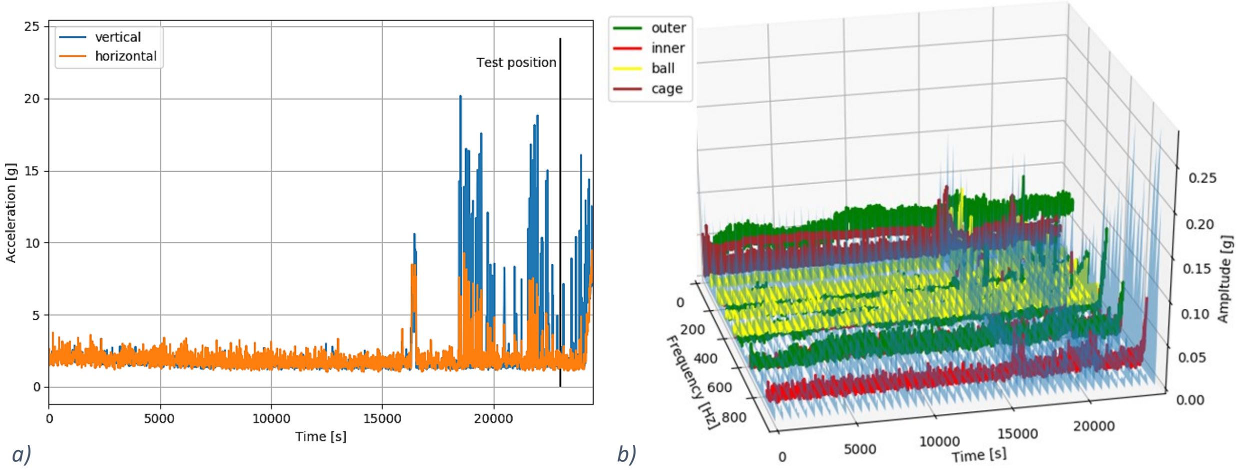

Figure 59: The sensor values of bearing dataset 1_6 over time: Figure a) shows the maximum acceleration value of the horizontal and vertical accelerators of each measurement record. In addition, the test position is marked. Figure b) shows the values of the horizontal sensor in the time-frequency domain. For this purpose, an FFT was performed for the measured values every 500 seconds. A) and b) do not show a special characteristic at the test position after 23,020 seconds. As can be seen in a) and b), there are increased measured values before and after the test position but not at the test position. In addition, a high degree of scattering of the measured values is recognizable [112].

Figure 59 illustrates that, as with bearing 2_5, there are no signs of a monotonic degradation at the test position: Neither in the characteristic frequencies in the time-frequency domain nor in the time domain.

It is worth to note that the measurement reveals a few short-duration high peaks from unknown source just before the test position. In contrast to bearing 1_4 and bearing 2_5, there is also an additional strong scattering of the accelerations throughout the bearing’s lifetime.

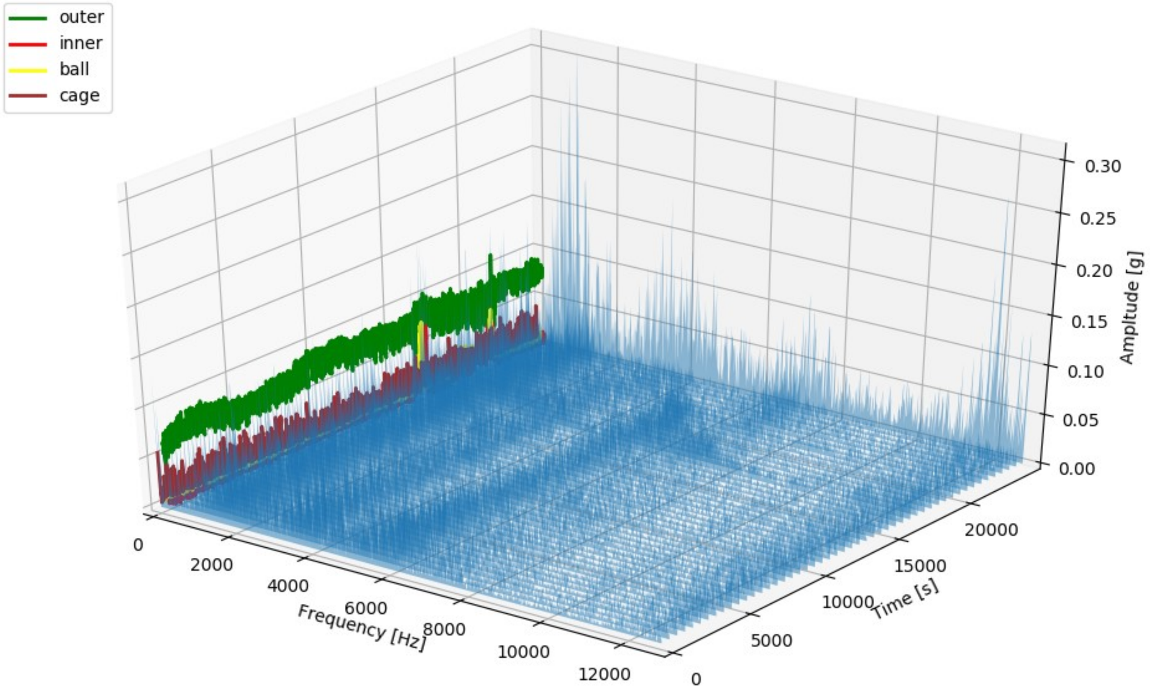

Figure 60: The amplitudes of the entire frequency range (up to 12,800 Hz) of the horizontal accelerometer of bearing dataset 1_6 over time. This plot shows hardly any increase in the amplitude of the fault frequencies. However, an increased amplitude of the frequency ranges of 1,000 Hz, 6,000 Hz, and 12,000 Hz is recognizable towards the end.

As already seen in the analyses of the other two bearings, Figure 60 again shows that the supposed natural frequencies at frequency ranges around 1,000 Hz, 6,000 Hz, and 12,000 Hz have an increasing trend. At the test position itself, all three frequency ranges are already excited. The noise over the entire frequency range, which has already been seen in Figure 59, is also evident here.

7.4.4.5 Assessment of the Analysis

The analysis of the reference dataset (bearing 1_4) shows a wear-out, as expected. The bearing has constant acceleration values up to the point when the degradation starts. From that point on, the measured acceleration values increase over time. This behavior is visible in the time domain as well as in the frequency domain in the range of the characteristic fault frequencies. In addition, the test position is inside the degradation area of the characteristic fault frequencies (degradation stage 3).

This is different from the measurements of the two outlier datasets. Here, in both cases, the test position shows neither increased acceleration values in the time domain nor increased amplitudes of the fault frequencies. However, there are increased amplitudes in specific frequency ranges such as 1,000 Hz and 6,000 Hz. Since all three examined bearings have an increasing trend of amplitudes in these ranges towards the end of their lifetime, this strongly indicates that these frequencies are the natural frequencies of the bearing components. Therefore, the test positions of the two outlier datasets are in degradation stage 2 rather than degradation stage 3.

This analysis confirms the constraints given in Section 6.3 that the degradation process must be at least in degradation stage 3. As can be seen in the three analyses above, the amplitudes of the fault frequencies are almost identical for every position before the beginning of this stage. Therefore, the LSTM can only determine the RUL value based on the absolute amplitude and not their correlation. However, as can be seen in the analyses above, the amplitudes before the beginning of degradation stage 3 differ from bearing to bearing even though they are in a healthy condition. Therefore, it can be assumed that the neural network, which does not have any information on the current lifetime, cannot differentiate between the individual positions in degradation stage 2. Therefore, it is only possible to determine the RUL if a degradation in the low-frequency range of the characteristic fault frequencies (degradation stage 3) is already taking place.

7.4.5 Conclusion

The performed benchmark has shown excellent results by using the presented RUL approach, in combination with transfer learning by using only low-frequency features. For transfer learning, a dataset of a completely different bearing can even be used. The achieved results are even superior to the winners of the IEEE PHM 2012 Data Challenge. Since the introduced approach is not using any frequencies above 900 Hz, this is particularly noteworthy. The estimated and the actual RUL are close for most of the test datasets used. However, two datasets had large deviations. This is due to the limitations of this approach explained in Section 6.3. This approach relies on the characteristic fault frequencies used by the intermediate domain and has therefore the best results for the RUL estimation of bearings in degradation stages 3 and 4. The test positions of the two outlier datasets are in degradation stage 2. Accordingly, the RUL estimation should only be used in a real-world scenario using this method if a degradation is already apparent. A realistic RUL value can only be estimated after that. The few other recent approaches that achieve a better PHM score, like Zhang et al. [157], utilize input in the time-frequency domain. Their RUL estimation approaches take the natural frequencies of the bearings into account, by using the entire bandwidth of the available frequencies. These frequencies are supposed to be at 1,000 . Certain research works especially concentrate on these high-frequency bands, such as the one of Yoo and Baek [196], which is presently the second-best approach with a PHM score of 0.62. Although the intermediate domain does not cover these frequencies, the frequency ranges of 6,000 Hz and 12,000 Hz cannot be captured with the industrial triaxial sensors currently available either. This kind of sensors, which are in the subject of this approach, frequently have a maximum sampling rate of around 5000 Hz (see Section 3.2.5). Therefore, this is also a constraint of the targeted sensors.

Based on the findings presented above, this benchmark can confirm the RUL solution from Chapter 6, which is also the answer to RQ2 that asks for a new RUL method, which can take benefits of a dataset of a different bearing type for a labeled target dataset that is recorded with sensors with low sampling rates.

7.5 Conclusion

The three case studies in this chapter have successfully verified the intermediate domain and the classification and RUL transfer learning approaches for predictive maintenance.

The exploration of the classification approach (Section 7.2) demonstrates the superiority of the LMMD in combination with the intermediate domain over the state-of-the-art methods used. In the semi-supervised transfer learning case of real bearing defects with strongly different process conditions (rotational speed) in the target domain, an increased accuracy of about 20.1% is achieved (87.7% for intermediate domain and LMMD vs. 67.6% for S-transform and Wasserstein). This approach was also successful in the direct benchmark with a state-of-the-art approach in Section 7.3. Here, an increase in the accuracy from 64.2% to 66.85% was achieved, although the approach in this benchmark cannot apply its full capabilities, such as support for different process conditions. Therefore, in response to RQ1, the presented approach is successfully validated in both case studies.

The benchmark for the RUL application, which is based on the IEEE PHM 2012 data challenge, also confirmed the suitability of the presented RUL approach (Section 7.4). Furthermore, this transfer learning approach, which is optimized for using a restricted frequency spectrum, showed even better results than the winning approaches of the challenge. Thus, the approach for RQ2 is also validated.

These case studies also verify the intermediate domain itself, which is the answer to RQ3. The exploration of Section 7.2 showed its supportive effect for transfer learning solutions in direct comparison to other sensor signal techniques. Furthermore, the three case studies in this chapter showed that the intermediate domain is stable enough to be used in different setups without any modifications.

In addition, the case studies also provide answers to RC1 (appropriate methods), RC2 (constraints), and RC3 (combinations and optimizations). This becomes especially visible in Section 7.2, where possible combinations of the appropriate methods, feature extraction, and transfer learning loss function are extensively combined in a practical use case. The different accuracies of the different methods show their useable application areas.

8 Conclusion and Outlook

8.1 Discussion

8.1.1 Introduction

As introduced in Section 1.1, one of the most demanded areas in predictive maintenance is the area of bearings. In this area, both RUL and classification are of interest. Furthermore, there is often a lack of usable bearing datasets to perform a reliable analysis. Accordingly, transfer learning is of particular interest for predictive maintenance tasks of bearings. For this reason, this thesis discussed the topic of "Transfer Learning for Predictive Maintenance Solutions" using the example of bearings. This chapter summarizes the answers to the specific research questions and the more general research challenges of Section 1.2.

8.1.2 Research Questions

During this thesis, three RQs were identified. They were subsequently answered in Chapter 4 (RQ3), Chapter 5 (RQ1), and Chapter 6 (RQ2). The answers to each question can be summarized in chronological order as follows:

RQ3: What are the necessary characteristics of a feature extraction method that is well suited for transfer learning? This method must be stable enough to be used on different bearing types without changing its parametrization or making significant changes to a subsequent machine learning model for different bearing types.

Nowadays, purely data-driven approaches are often used for deep learning. It is advisable to choose a hybrid approach to increase accuracy compared to existing approaches. For this purpose, a new intermediate domain that considers context parameters in the form of the characteristic frequencies of rotating machine components was developed in Chapter 4. This intermediate domain was explained using the example of the fault classification of bearings and can be used for two things: First, it is a normal feature extraction method that can be used for different machine learning approaches such as CNNs or LSTMs. However, by considering the characteristic frequencies of rotating components, it also automatically performs a kind of transfer learning. This is achieved by minimizing the difference between different types of the same component and different process parameters by creating separate layers for each characteristic frequency, thus leading to better transferability to another domain. An exploration also verifies this assumption in Section 7.2, where the usage of a pretrained network of a source domain results in better accuracies for a target domain than other techniques, such as HHT, without any training with target domain data.

The focus on the fault frequency also leads to a stable solution that can be used on different types without any modifications. This has been verified in the different case studies of Chapter 7. In each of them, the approach that uses the intermediate domain is the best approach for the given use case. This result is reached without any modifications of the intermediate domain parameters.

RQ1: What are the necessary characteristics of a new classification method, which can take benefits of a dataset of a different bearing type for a partly labeled target dataset that is collected under different process conditions?

As already mentioned in the answer to RQ3, a hybrid approach can increase the accuracy of deep learning solutions. Therefore, the proposed intermediate domain of Chapter 4 is used to preprocess the input data of the presented classification approach. This data is later used as input for a CNN. The CNN itself consists of doubled convolutional layers, which improve the significance of the features through an additional nonlinearity (see Section 5.4). For the transfer learning task, a new function for calculating the transfer loss has been developed based on the newly developed intermediate domain (see Section 5.5). This is called Layered Maximum Mean Discrepancy because the layers of the intermediate domain are used. In addition, MMD is used to calculate the discrepancy of each layer. The sum of the functions is then used as a loss function during transfer learning. Therefore, like the intermediate domain, it is also a hybrid approach. This approach has been evaluated in Section 7.2 with the help of three different datasets. This new approach outperforms the existing state-of-the-art transfer learning techniques by increasing the accuracy by about 20.1%. In addition, a direct benchmark to another research work based on the same test setup has been performed in Section 7.3. This benchmark uses similar bearing types and process conditions in the source and the target domain, meaning that the advantages of the proposed approach cannot be fully exploited. Nevertheless, it still outperforms the other research by an accuracy of 2.6%.

RQ2: What are the necessary characteristics of a new RUL method, which can take benefits of a dataset of a different bearing type, for a labeled target dataset that is recorded with sensors with low sampling rates?

This research question is related to RQ1 and RQ3 in that the RUL approach extends the usage of sensor data of the classification approach with time dependencies. Therefore, the basic concept of using a hybrid approach that extracts valuable features into an intermediate domain is valid for both (see Section 6.3). However, before using the features of the intermediate domain for the regression task of an RUL estimation, they must be extracted. As already verified for the classification task, an appropriate approach is to use convolutional layers. Therefore, a second machine learning-based feature extraction layer is used (Section 6.4.2). This layer has the same layout as the convolutional layers in the classification approach. For the RUL estimation itself, the use of an LSTM has been shown to be appropriate. This decision is based on the current state of the art in Section 3.3 and on the derivation/evaluation in Section 6.4.

In addition to being used for the feature extraction, using the intermediate domain in combination with the convolutional layers has a significant advantage when performing transfer learning. As described in the answer to RQ3, the intermediate domain reduces the difference between the source and the target domain, which leads to a better transferability. In addition, through the convolutional layer-based feature extraction, it is possible to perform network-based transfer learning by pre-training and freezing this layer with a dataset of another component. These source domain datasets are not limited to RUL datasets. As evaluated in Section 6.4.4, even a classification dataset can be used. This approach has been evaluated on an IEEE PHM 2012 data challenge-based benchmark in Section 7.4. It outperforms the winning approach of this challenge with a PHM score of 0.35 versus 0.3066. A few other current approaches provide better results than the presented approach. However, these are not directly comparable since the proposed approach intentionally uses only a limited feature space to cover the intended industrial use of triaxial sensors, which have only a low sampling frequency.

8.1.3 Research Challenges

In addition to the specific and measurable research questions, this thesis has also contributed content to the three RCs introduced in Section 1.2. These RCs are more general than the RQs; therefore, a complete answer is not possible. However, the following input was given to address these challenges:

RC1: Which methods are appropriate for predictive maintenance tasks of machines based on features of sensor data?

With the rise of machine learning over the last decade, a lot of research has been performed on the topic of predictive maintenance. Therefore, the current state of the art covers manifold methods. However, open questions are related to their usage for real life predictive maintenance scenarios.

- Are the methods appropriate for the analysis of sensor data?
- Which feature extraction methods are suitable for the needs of predictive maintenance?
- Which deep learning methods are available for this use case?

RC2: Under which constraints can the different methods be used?

This question is especially important for the different predictive maintenance scenarios that are also based on a variety of different dataset types. On the one hand, amongst others the following questions arise regarding feature extraction:

- Which methods are appropriate for stationary signals?
- Which are even usable with nonlinear and non-stationary signals?

On the other hand, different machine learning methods exist. Here, relevant questions include:

- Which machine learning methods are well suited for small training datasets?
- Which machine learning methods are only usable for large datasets?
- Which methods are suited for transfer learning to overcome the problem of small datasets?

The first two research challenges are directly related and can be answered together. As shown in Figure 61, there are two relevant types of methods for predictive maintenance based on sensor data. The first is the feature extraction method, which is used to extract more powerful indicators (features) from the raw sensor data and to reduce the amount of data. The second is the algorithm, which uses the extracted features to perform the predictive maintenance task. In the case of transfer learning, a third type is added, which is the transfer learning method itself. These three methods are discussed in greater detail below.

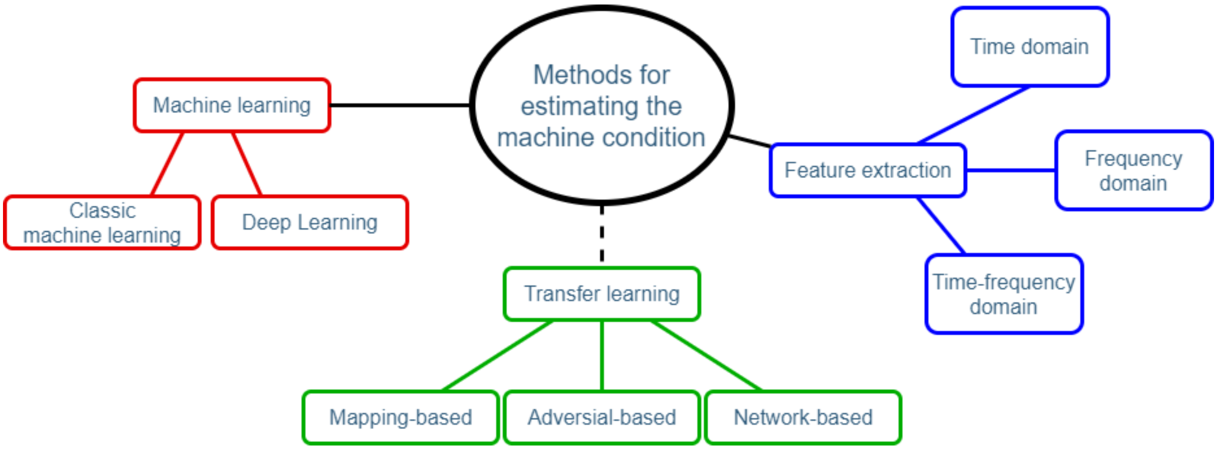

Figure 61: Different methods for estimating the machine condition and their interactions. There are two major types of methods: feature extraction and machine. In addition, an optional third method is the used transfer learning method. Here the most common deep learning methods are shown.

Feature Extraction Methods:

As stated in Section 2.3, feature extraction methods for sensor data can be distinguished into three types (time domain, frequency domain, and time-frequency domain). Two important conditions need to be considered when choosing a feature extraction method. First, it is necessary to have an understanding of the data to decide which feature extraction method to use. For instance, it is essential to know if the data is non-stationary and nonlinear. In this case, simple time domain and frequency domain analyses are impossible. Instead, it would be necessary to use techniques like HHT or S-transform. However, often this is not the case, and less complex and less computation time-intensive algorithms like STFT or even only time-domain features can be used. This is also true for the presented use case of predictive maintenance of bearings. Since the rotational speed during test runs can be kept constant, there is no need to use approaches like an S-transform. Therefore, the presented solutions for predictive maintenance of bearings (see Chapter 5 and Chapter 6) can use the presented intermediate domain of Chapter 4, which is comparable to an STFT in terms of time-frequency behavior. The predictive maintenance algorithm is the second consideration when selecting a feature extraction method. Classical machine learning algorithms like SVMs can only handle a few features. Therefore, time-domain features like the root mean square are often used. By using deep learning algorithms like CNNs, more detailed features, such as images, can be used.

Machine Learning Methods:

The selection of the machine learning method is another important decision. As described in Section 2.2.1 and Section 3.2.6 using machine learning for predictive maintenance is superior to non-machine learning approaches, especially in complex scenarios where much fine-tuning would be needed. Many predictive maintenance approaches use classic machine learning methods like SVMs. Nevertheless, some challenges are harder to master with traditional machine learning only. This is, for instance, when there is only a small amount of training data, which may be partly or even completely unlabeled. In this context, deep learning methods like CNNs are more suitable. This is based on the possibility of easily using (deep) transfer learning to transfer knowledge from a well-known source dataset to a small and unlabeled target dataset (see Section 2.4).

Transfer Learning Methods:

Using transfer learning leads to another degree of freedom for the decision of the used methods: In this case, the transfer function is also of interest (see Section 2.4.3.5). The most common transfer learning methods for deep learning are network-based deep transfer learning, mapping-based deep transfer learning, and adversarial-based deep learning. Network-based and mapping-based approaches show promising results for the use-case of predictive maintenance of bearings (see Sections 3.3.2 and 3.3.3). These approaches can transfer parts of a pre-trained network of the source domain to the target domain. However, mapping-based approaches, which measure the probability distance of the target and the source domain, such as MMD, also show very promising results. Here again, different existing methods are available (see Section 2.4.6).

Different methods for feature extraction and transfer learning and their constraints on small labeled, partly labeled, or unlabeled datasets have been shown in the case studies presented in Chapter 7. The number of possible combinations of different methods leads directly to RC3.

RC3: How can existing methods be combined and optimized to complement each other?

There are many different methods, but it remains an open question how they can be used together to achieve optimal results. The combination of different feature extraction methods and transfer learning methods is of special interest here.

- Are data-driven feature extraction approaches like the Hilbert-Huang transform or hybrid approaches, such as handcrafted intermediate domains, better suited for this task?
- Based on the often-small datasets for a specific machine component, can transfer learning be a solution?
- Is it possible to use such a solution even for partly or unlabeled datasets?

This challenge can be described with the help of predictive maintenance tasks for bearings. Considering that a combination of CNN and transfer learning is used for bearing fault classification, there are degrees of freedom in the feature extraction method and the transfer learning function (see Figure 61).

Feature Extraction:

If a classical CNN is assumed, the input must be a matrix, as it is when using an image. For sensor signals, it is therefore common to transform the sensor data into the time-frequency domain. This transformation automatically gives a two-dimensional input for a CNN. A positive side effect of this transformation is that non-stationary signals can also be analyzed. This can be achieved, among others, by an S-transformation or an HHT. These are purely data-driven approaches. In addition, it is also possible to use hybrid approaches, which make use of the properties of the datasets and generate an intermediate domain and therefore improve the accuracy of the machine learning task (see Chapter 4).

Transfer Learning:

For transfer learning, there are several distance-based transfer functions available, such as MMD, CORAL, MK-MMD, and Wasserstein. Since both the feature extraction method and the transfer function can be easily exchanged, it is possible to benchmark them directly. This was done while evaluating the classification approach, which was developed in Chapter 5 to answer RQ1 in the case studies in Section 7.2. These case studies have also shown that his approach is usable for partly and unlabeled data of the target domain. In addition, network-based transfer learning is also an option for deep learning approaches since, for these approaches, parts of the trained network can be transferred to the target domain. This has been proposed for the solution for the classification approach in Sections 5.4.3 and 5.4.4 (RQ1) as well as for the RUL approach in Section 6.4.4 (RQ2). In addition, there is also the possibility of using an intermediate domain like the one proposed in Chapter 4. Such an intermediate domain can also be a kind of transfer learning.

Machine Learning Method:

The combination of existing methods can also be a combination of different machine learning methods in the machine learning part. Especially for the estimation of the RUL, two machine learning-based methods combined in one approach are a valid option. One machine learning part can be used for the feature extraction while the other can be used for the RUL estimation itself. For instance, convolutional layers can be used for the feature extraction of the intermediate domain and can optimize the feature extraction by training of the neuronal network. The output of this network can then be used as input for a second machine learning part for the RUL estimation. This approach is used for the answer of RQ2 in Chapter 6.

In summary, a transfer machine learning process consists of feature extraction, the transfer learning algorithm, and the machine learning algorithm. Each can act as a single method or a combination of several different ones, depending on the particular application.

8.1.4 Conclusion

To sum up, all research questions of this thesis have been answered successfully with the help of a new intermediate domain and solutions for classification and a remaining useful life task for bearings. Both solutions rely on transfer learning in the context of predictive maintenance. Those solutions are easy to implement and lead to better results than existing approaches by using a new hybrid approach. The two solutions and the intermediate domain have already been published in research papers [112, 133]. In addition, a survey paper on the state-of-the-art for predictive maintenance of bearings has also been published [54].

In addition to answering the research questions, valuable content was provided to address the RC. This has been done with the solution for the RQs and also during the introduction and presentation of the state of the art in Chapter 2 and Chapter 3.

8.2 Outlook

The previous discussion has outlined the improvements in predictive maintenance, which can be achieved using transfer learning. However, further improvements to the presented transfer learning approaches are still possible.

8.2.1 Transfer Learning for Classification

The presented classification solution was developed to provide a universal solution to different bearing types. Leaving this out of consideration, several approaches can be pursued. The first is to carefully investigate which lambdas (weights of the different loss functions) are best suited for a specific application. This contradicts the generic approach, which can be applied directly without modification. However, it might bring about improvements in classification accuracy.

Another option is to vary the number of frozen layers or the learning rate. In the context of this thesis, one example examined how far the number of frozen layers influences the result and found no significant difference. However, the difference may be more significant for other datasets.

In addition to modifications in the machine learning model, the input may also be modified. Here, the width of the frequency band and image size of the intermediate domain are points that may be adjusted.

8.2.2 Transfer Learning for Remaining Useful Life

The RUL approach is an extension of the classification approach. Therefore, the considerations of the previous chapter for the classification approach can also be considered for the RUL approach. However, there are also possible improvements that specifically address the requirements related to RUL. These include an optimization of the degradation model towards a nonlinear behavior or an optimization of the window size.

However, as shown in the benchmark for the RUL approach (Section 7.4), the most promising improvement would be the integration of the excited natural frequencies of the components. For this purpose, a natural frequency layer would have to be added to the intermediate domain. Unfortunately, this approach comes with two downsides. The first is that the typically available classification dataset cannot be used as a source domain dataset since the natural frequencies cannot be estimated in a static classification dataset. Another disadvantage is that the use of triaxial accelerometers would no longer be possible because their frequency range would not be sufficient to measure the natural frequencies. In order to prove this possibility in the future, matching datasets are currently being collected.

References

[1] J. Fernandes, J. Reis, N. Melão, L. Teixeira, and M. Amorim, “The Role of Industry 4.0 and BPMN in the Arise of Condition-Based and Predictive Maintenance: A Case Study in the Automotive Industry,” *Applied Sciences*, vol. 11, no. 8, p. 3438, 2021, doi: 10.3390/app11083438.

[2] “Predictive Maintenance setzt sich durch,” *Design & Elektronik*, no. 3, p. 10, 2021.

[3] J. Fleischer, M. Schopp, A. Broos, and J. Wieser, “Sustainable Design of Machine Tools through Load-Dependent Interventions and Adapted Services,” in *Manufacturing Systems and Technologies for the New Frontier: The 41st CIRP Conference on Manufacturing Systems May 26–28 2008 Tokyo Japan*, 2008, pp. 173–176.

[4] GAM, *A better way to protect machine spindles from collisions.* [Online]. Available: https://www.gamweb.com/documents/SpindleSafetySystemArticle-DesignWorld-September2012.pdf (accessed: Apr. 11 2018).

[5] P. K. Kankar, S. C. Sharma, and S. P. Harsha, “Fault diagnosis of ball bearings using continuous wavelet transform,” *Applied Soft Computing*, vol. 11, no. 2, pp. 2300–2312, 2011, doi: 10.1016/j.asoc.2010.08.011.

[6] A. Géron, *Hands-on machine learning with Scikit-Learn and TensorFlow: Concepts, tools, and techniques to build intelligent systems*. Sebastopol, CA: O'Reilly Media, 2017.

[7] J. Patterson and A. Gibson, *Deep learning: A practitioner's approach*. Sebastopol, CA: O'Reilly Media, 2017. [Online]. Available: http://proquest.tech.safaribooksonline.de/9781491924570

[8] S. Zhang, S. Zhang, B. Wang, and T. G. Habetler, “Machine Learning and Deep Learning Algorithms for Bearing Fault Diagnostics - A Comprehensive Review,” Jan. 2019. [Online]. Available: http://arxiv.org/pdf/1901.08247v1

[9] S. H. Bang, R. Ak, A. Narayanan, Y. T. Lee, and H. Cho, “A survey on knowledge transfer for manufacturing data analytics,” *Computers in Industry*, vol. 104, pp. 116–130, 2019, doi: 10.1016/j.compind.2018.07.001.

[10] A. L. Samuel, “Some Studies in Machine Learning Using the Game of Checkers,” *IBM J. Res. & Dev.*, vol. 3, no. 3, pp. 210–229, 1959, doi: 10.1147/rd.33.0210.

[11] Y. Roh, G. Heo, and S. E. Whang, “A Survey on Data Collection for Machine Learning: a Big Data -- AI Integration Perspective,” Nov. 2018. [Online]. Available: http://arxiv.org/pdf/1811.03402v2

[12] H. Cheng, X. Kong, G. Chen, Q. Wang, and R. Wang, “Transferable convolutional neural network based remaining useful life prediction of bearing under multiple failure behaviors,” *Measurement*, vol. 168, 108286, 2021, doi: 10.1016/j.measurement.2020.108286.

[13] L. Liao and F. Köttig, “A hybrid framework combining data-driven and model-based methods for system remaining useful life prediction,” *Applied Soft Computing*, vol. 44, pp. 191–199, 2016, doi: 10.1016/j.asoc.2016.03.013.

[14] X. Su, H. Liu, L. Tao, C. Lu, and M. Suo, “An end-to-end framework for remaining useful life prediction of rolling bearing based on feature pre-extraction mechanism and deep adaptive transformer model,” *Computers & Industrial Engineering*, vol. 161, p. 107531, 2021, doi: 10.1016/j.cie.2021.107531.

[15] M. E. Taylor and P. Stone, “Transfer Learning for Reinforcement Learning Domains: A Survey,” *Journal of Machine Learning Research*, vol. 10, no. 56, pp. 1633–1685, 2009, doi: 10.1145/1577069.1755839.

[16] Alexander Gepperth and Barbara Hammer, “Incremental learning algorithms and applications,” [Online]. Available: https://hal.archives-ouvertes.fr/hal-01418129/file/article.pdf

[17] D. Sarkar, R. Bali, and T. Ghosh, *Hands-On Transfer Learning with Python: Implement Advanced Deep Learning and Neural Network Models Using TensorFlow and Keras*. Birmingham: Packt Publishing Ltd, 2018. [Online]. Available: https://ebookcentral.proquest.com/lib/gbv/detail.action?docID=5507771

[18] J. Schmidhuber, “Deep learning in neural networks: An overview,” *Neural networks : the official journal of the International Neural Network Society*, vol. 61, pp. 85–117, 2015, doi: 10.1016/j.neunet.2014.09.003.

[19] Y. LeCun *et al.,* “Handwritten digit recognition with a backpropagation network,” *Advances in Neural Information Processing Systems*, pp. 396–404, 1990.

[20] H. Lee, P. Pham, Y. Largman, and A. Y. Ng, “Unsupervised feature learning for audio classification using convolutional deep belief networks,” in *Advances in Neural Information Processing Systems Conference*, 2009, pp. 1096–1104.

[21] T. N. Sainath, A. Mohamed, B. Kingsbury, and B. Ramabhadran, “Deep convolutional neural networks for LVCSR,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Vancouver, BC, Canada, May. 2013 - May. 2013, pp. 8614–8618.

[22] X. Chen, S. Xiang, C.-L. Liu, and C.-H. Pan, “Vehicle Detection in Satellite Images by Hybrid Deep Convolutional Neural Networks,” *IEEE Geosci. Remote Sensing Lett.*, vol. 11, no. 10, pp. 1797–1801, 2014, doi: 10.1109/LGRS.2014.2309695.

[23] Y. LeCun, U. Muller, J. Ben, E. Cosatto, and B. Flepp, “Off-Road Obstacle Avoidance through End-to-End Learning,” in *Proceedings of the 18th International Conference on Neural Information Processing Systems*, 2005, pp. 739–746.

[24] L. Deng, O. Abdel-Hamid, and D. Yu, “A deep convolutional neural network using heterogeneous pooling for trading acoustic invariance with phonetic confusion,” in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013, pp. 6669–6673.

[25] P. Sermanet, S. Chintala, and Y. LeCun, “Convolutional Neural Networks Applied to House Numbers Digit Classification,” Apr. 2012. [Online]. Available: https://arxiv.org/pdf/1204.3968

[26] L. Wen, X. Li, L. Gao, and Y. Zhang, “A New Convolutional Neural Network Based Data-Driven Fault Diagnosis Method,” *IEEE Trans. Ind. Electron.*, pp. 5590–5998, 2017, doi: 10.1109/TIE.2017.2774777.

[27] A. G. Parth Goel, “A survey on Deep Transfer Learning for Convolution Neural Networks,” *International Journal of Advanced Science and Technology*, vol. 29, no. 06, pp. 8399–8410, 2020. [Online]. Available: http://sersc.org/journals/index.php/IJAST/article/view/25284

[28] A. P. Wibawa, A. B. P. Utama, H. Elmunsyah, U. Pujianto, F. A. Dwiyanto, and L. Hernandez, “Time-series analysis with smoothed Convolutional Neural Network,” *Journal of big data*, vol. 9, no. 1, p. 44, 2022, doi: 10.1186/s40537-022-00599-y.

[29] C. Baru, Ed., *2019 IEEE International Conference on Big Data: Dec 9-Dec 12, 2019, Los Angeles, CA, USA : proceedings*. Piscataway, NJ, USA: IEEE, 2019. [Online]. Available: http://ieeexplore.ieee.org /servlet/opac?punumber=8986695

[30] B. Sahoo, *Data-Driven Remaining Useful Life (RUL) Prediction.* [Online]. Available: https:// biswajitsahoo1111.github.io/rul_codes_open/

[31] Niousha Rasifaghihi, *Predictive Analytics: Regression Analysis with LSTM, GRU and BiLSTM in TensorFlow.* [Online]. Available: https://towardsdatascience.com/predictive-analysis-rnn-lstm-and-gru-to-predict-water-consumption-e6bb3c2b4b02 (accessed: Dec. 17 2020).

[32] M. Abboush, D. Bamal, C. Knieke, and A. Rausch, “Intelligent Fault Detection and Classification Based on Hybrid Deep Learning Methods for Hardware-in-the-Loop Test of Automotive Software Systems,” *Sensors (Basel, Switzerland)*, vol. 22, no. 11, 2022, doi: 10.3390/s22114066.

[33] I. El-Thalji and E. Jantunen, “A summary of fault modelling and predictive health monitoring of rolling element bearings,” *Mechanical Systems and Signal Processing*, 60-61, pp. 252–272, 2015, doi: 10.1016/j.ymssp.2015.02.008.

[34] S. Kang, D. Ma, Y. Wang, C. Lan, Q. Chen, and V. I. Mikulovich, “Method of assessing the state of a rolling bearing based on the relative compensation distance of multiple-domain features and locally linear embedding,” *Mechanical Systems and Signal Processing*, vol. 86, pp. 40–57, 2017, doi: 10.1016/j.ymssp.2016.10.006.

[35] H. Kronmüller, Ed., *Digitale Signalverarbeitung: Grundlagen, Theorie, Anwendungen in der Automatisierungstechnik*. Berlin, Heidelberg: Springer, 1991.

[36] C. Lessmeier, J. K. Kimotho, D. Zimmer, and W. Sextro, “Condition Monitoring of Bearing Damage in Electromechanical Drive Systems by Using Motor Current Signals of Electric Motors: A Benchmark Data Set for Data-Driven Classification,” Bilbao, Spain, 07.2016.

[37] S. J. Lacey, “An Overview of Bearing Vibration Analysis,” *maintenance & asset management*, vol. 2008, no. 23.

[38] M. Werner, “Schnelle Fouriertransformation (FFT),” in *Studium Technik, Digitale Signalverarbeitung mit MATLAB: Ein Praktikum mit 16 Versuchen*, M. Werner, Ed., 1st ed., Braunschweig: Vieweg, 2001, pp. 47–56.

[39] M. Unal, M. Onat, M. Demetgul, and H. Kucuk, “Fault diagnosis of rolling bearings using a genetic algorithm optimized neural network,” *Measurement*, vol. 58, pp. 187–196, 2014, doi: 10.1016/j.measurement.2014.08.041.

[40] H. Hanselka, S. Herold, and R. Nordmann, “Schwingungen,” in *Dubbel*, K.-H. Grote and J. Feldhusen, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 1002–1021.

[41] R. G. Stockwell, “Why use the S-Transform?,” *American Mathematical Society Pseudo-differential equations and time frequency analysis*, vol. 52, 2007.

[42] A. Stepchenko, J. Chizhov, L. Aleksejeva, and J. Tolujew, “Nonlinear, Non-stationary and Seasonal Time Series Forecasting Using Different Methods Coupled with Data Preprocessing,” *Procedia Computer Science*, vol. 104, pp. 578–585, 2017, doi: 10.1016/j.procs.2017.01.175.

[43] C. Cheng *et al.,* “Time series forecasting for nonlinear and non-stationary processes: a review and comparative study,” *IIE Transactions*, vol. 47, no. 10, pp. 1053–1071, 2015, doi: 10.1080/0740817X.2014.999180.

[44] B. Rajoub, “Characterization of biomedical signals: Feature engineering and extraction,” in *Biomedical Signal Processing and Artificial Intelligence in Healthcare*: Elsevier, 2020, pp. 29–50.

[45] W. Yang, C. Little, and R. Court, “S-Transform and its contribution to wind turbine condition monitoring,” *Renewable Energy*, vol. 62, pp. 137–146, 2014, doi: 10.1016/j.renene.2013.06.050.

[46] C M Leavey, M N James, J Summerscales and R Sutton, “An introduction to wavelet transforms: a tutorial approach: a tutorial approach,” 2002.

[47] N. O. Attoh-Okine and N. E. Huang, *The Hilbert-Huang transform in engineering*. New York: Taylor & Francis, 2005.

[48] H. Dong, K. Qi, X. Chen, Y. Zi, Z. He, and B. Li, “Sifting process of EMD and its application in rolling element bearing fault diagnosis,” *J Mech Sci Technol*, vol. 23, no. 8, pp. 2000–2007, 2009, doi: 10.1007/s12206-009-0438-9.

[49] N. E. Huang and Z. Wu, “A review on Hilbert-Huang transform: Method and its applications to geophysical studies,” *Rev. Geophys.*, vol. 46, no. 2, L13705, 2008, doi: 10.1029/2007RG000228.

[50] Victor, *Introduction to the Empirical Mode Decomposition Method.* [Online]. Available: https://www.mql5.com/en/articles/439 (accessed: Oct. 4 2018).

[51] Z. K. Peng, P. W. Tse, and F. L. Chu, "An improved Hilbert–Huang transform and its application in vibration signal analysis," *Journal of Sound and Vibration*, vol. 286, 1-2, pp. 187–205, 2005, doi: 10.1016/j.jsv.2004.10.005.

[52] R. G. Stockwell, L. Mansinha, and R. P. Lowe, "Localization of the complex spectrum: the S transform," *IEEE Trans. Signal Process.*, vol. 44, no. 4, pp. 998–1001, 1996, doi: 10.1109/78.492555.

[53] M. Zhao, B. Tang, and Q. Tan, "Bearing remaining useful life estimation based on time–frequency representation and supervised dimensionality reduction," *Measurement*, vol. 86, pp. 41–55, 2016, doi: 10.1016/j.measurement.2015.11.047.

[54] S. Schwendemann, Z. Amjad, and A. Sikora, "A survey of machine-learning techniques for condition monitoring and predictive maintenance of bearings in grinding machines," *Computers in Industry*, vol. 125, 103380, 2021, doi: 10.1016/j.compind.2020.103380.

[55] S. J. Pan and Q. Yang, "A Survey on Transfer Learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, 2010, doi: 10.1109/TKDE.2009.191.

[56] C. Giraud-Carrier, "A Note on the Utility of Incremental Learning," *AI Commun*, vol. 13, pp. 215–224, 2000.

[57] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, "A Survey on Deep Transfer Learning," Aug. 2018. [Online]. Available: http://arxiv.org/pdf/1808.01974v1

[58] K. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey of transfer learning," *J Big Data*, vol. 3, no. 1, 2016, doi: 10.1186/s40537-016-0043-6.

[59] C.-W. Seah, Y.-S. Ong, and I. W. Tsang, "Combating Negative Transfer From Predictive Distribution Differences," *IEEE transactions on cybernetics*, vol. 43, no. 4, pp. 1153–1165, 2013, doi: 10.1109/TSMCB.2012.2225102.

[60] Z. Wang, Z. Dai, B. Poczos, and J. Carbonell, "Characterizing and Avoiding Negative Transfer," in *CVPR 2019: Proceedings*, Long Beach, CA, USA, op. 2019, pp. 11285–11294.

[61] M. Wang and W. Deng, "Deep Visual Domain Adaptation: A Survey," Feb. 2018. [Online]. Available: http://arxiv.org/pdf/1802.03601v4

[62] Rui Xia, Chengqing Zong, Xuelei Hu, Erik Cambria, "Feature Ensemble Plus Sample Selection: Domain Adaptation for Sentiment Classification (Extended Abstract),"

[63] W. Dai, Q. Yang, G.-R. Xue, and Y. Yu, "Boosting for transfer learning," in *Proceedings of the 24th international conference on Machine learning*, Corvalis, Oregon, 2007, pp. 193–200.

[64] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, “A Kernel Two-Sample Test,” *Journal of Machine Learning Research*, pp. 723–773, 2012.

[65] M. Hamilton, “Semi-Supervised Translation with MMD Networks,” Oct. 2018. [Online]. Available: http://arxiv.org/pdf/1810.11906v1

[66] Y. Tang, B. Wu, L. Peng, and C. Liu, “Semi-Supervised Transfer Learning for Convolutional Neural Network Based Chinese Character Recognition,” in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, Kyoto, Nov. 2017 - Nov. 2017, pp. 441–447.

[67] J.-T. Huang, J. Li, D. Yu, L. Deng, and Y. Gong, “Cross-language knowledge transfer using multilingual deep neural network with shared hidden layers,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Vancouver, BC, Canada, May. 2013 - May. 2013.

[68] E. Tzeng, J. Hoffman, T. Darrell, and K. Saenko, “Simultaneous Deep Transfer Across Domains and Tasks,” in *2015 IEEE International Conference on Computer Vision: 11-18 December 2015, Santiago, Chile : proceedings*, Santiago, Chile, 2015, pp. 4068–4076.

[69] A. Karpathy, *Multi-Task Learning in the Wilderness.* [Online]. Available: https://slideslive.com/38917690/multitask-learning-in-the-wilderness (accessed: Oct. 14 2020).

[70] J. Roach, *What’s that? Microsoft’s latest breakthrough, now in Azure AI, describes images as well as people do.* [Online]. Available: https://blogs.microsoft.com/ai/azure-image-captioning/ (accessed: Oct. 14 2020).

[71] C. Szegedy *et al.,* “Going deeper with convolutions,” in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1–9.

[72] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell, “Deep Domain Confusion: Maximizing for Domain Invariance,” Dec. 2014. [Online]. Available: http://arxiv.org/pdf/1412.3474v1

[73] M. Ghifary, W. B. Kleijn, M. Zhang, D. Balduzzi, and W. Li, “Deep Reconstruction-Classification Networks for Unsupervised Domain Adaptation,” Jul. 2016. [Online]. Available: http://arxiv.org/pdf/1607.03516v2

[74] Y. Ganin *et al.,* “Domain-Adversarial Training of Neural Networks,” *Journal of Machine Learning Research 2016.* [Online]. Available: http://arxiv.org/pdf/1505.07818v4

[75] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, “Adversarial Discriminative Domain Adaptation,” Feb. 2017. [Online]. Available: http://arxiv.org/pdf/1702.05464v1

[76] F. Emmert-Streib, Z. Yang, H. Feng, S. Tripathi, and M. Dehmer, “An Introductory Review of Deep Learning for Prediction Models With Big Data,” *Frontiers in artificial intelligence*, vol. 3, p. 4, 2020, doi: 10.3389/frai.2020.00004.

[77] P. Gupta, P. Malhotra, L. Vig, and G. Shroff, "Transfer Learning for Clinical Time Series Analysis using Recurrent Neural Networks," Jul. 2018. [Online]. Available: http://arxiv.org/pdf/1807.01705v1

[78] S. Yoon, H. Yun, Y. Kim, G. Park, and K. Jung, "Efficient Transfer Learning Schemes for Personalized Language Modeling using Recurrent Neural Network," Jan. 2017. [Online]. Available: http://arxiv.org/pdf/1701.03578v1

[79] Toby Perrett and Dima Damen, "DDLSTM: Dual-Domain LSTM for Cross-Dataset Action Recognition,"

[80] Xiao Ding, Bibo Cai, Ting Liu, and Qiankun Shi, "Domain Adaptation via Tree Kernel Based Maximum Mean Discrepancy for User Consumption Intention Identification,"

[81] P. R. d. O. Da Costa, A. Akçay, Y. Zhang, and U. Kaymak, "Remaining useful lifetime prediction via deep domain adaptation," *Reliability Engineering & System Safety*, vol. 195, p. 106682, 2020, doi: 10.1016/j.ress.2019.106682.

[82] C. Liu and K. Gryllias, "Unsupervised Domain Adaptation based Remaining Useful Life Prediction of Rolling Element Bearings," in *Proceedings of the European Conference of the PHM Society 2020*, 2020.

[83] C. Cheng, B. Zhou, G. Ma, D. Wu, and Y. Yuan, "Wasserstein Distance based Deep Adversarial Transfer Learning for Intelligent Fault Diagnosis," Mar. 2019. [Online]. Available: http://arxiv.org /pdf/1903.06753v1

[84] M. Long, Y. Cao, J. Wang, and M. I. Jordan, "Learning Transferable Features with Deep Adaptation Networks," Feb. 2015. [Online]. Available: http://arxiv.org/pdf/1502.02791v2

[85] M. Long, H. Zhu, J. Wang, and M. I. Jordan, "Deep Transfer Learning with Joint Adaptation Networks," May. 2016. [Online]. Available: http://arxiv.org/pdf/1605.06636v2

[86] Y. He and G. Ding, "Deep Transfer Learning for Image Emotion Analysis: Reducing Marginal and Joint Distribution Discrepancies Together," *Neural Process Lett*, vol. 13, no. 5, p. 723, 2019, doi: 10.1007/s11063-019-10035-7.

[87] H. Yan, Y. Ding, P. Li, Q. Wang, Y. Xu, and W. Zuo, "Mind the Class Weight Bias: Weighted Maximum Mean Discrepancy for Unsupervised Domain Adaptation," May. 2017. [Online]. Available: http://arxiv.org/pdf/1705.00609v1

[88] J. Shen, Y. Qu, W. Zhang, and Y. Yu, "Wasserstein Distance Guided Representation Learning for Domain Adaptation," Jul. 2017. [Online]. Available: http://arxiv.org/pdf/1707.01217v4

[89] J. Solomon, R. Rustamov, L. Guibas, and A. Butscher, "Wasserstein Propagation for Semi-Supervised Learning," in *Proceedings of the 31st International Conference on Machine Learning*, 2014, pp. 306–314.

[90] B. Sun and K. Saenko, “Deep CORAL: Correlation Alignment for Deep Domain Adaptation,” Jul. 2016. [Online]. Available: http://arxiv.org/pdf/1607.01719v1

[91] *Was ist Industrie 4.0?* [Online]. Available: https://www.plattform-i40.de/PI40/Navigation/DE/Industrie40/WasIndustrie40/was-ist-industrie-40.html (accessed: Feb. 3 2021).

[92] M. Moore, “What is Industry 4.0? Everything you need to know,” *TechRadar pro*, 05 Nov., 2019. https://www.techradar.com/news/what-is-industry-40-everything-you-need-to-know (accessed: Feb. 3 2021).

[93] G. Schuh, R. Anderl, J. Gausemeier, M. ten Hompel, and W. Wahlster, “Industrie 4.0 Maturity Index.: Managing the Digital Transformation of Companies,” 2017.

[94] Nagdev Amruthnath and Tarun Gupta, “Fault Class Prediction in Unsupervised Learning using Model-Based Clustering Approach,” 2018.

[95] H. Opitz, R. Piekenbrink, and K. Honrath, *Untersuchungen an Werkzeugmaschinenelementen*. Wiesbaden, s.l.: VS Verlag für Sozialwissenschaften, 1959.

[96] O. Schenk and H. Pittroff, *Das Schwingungsverhalten des Systems Spindel und Wälzlager: Berechnung der Starrheit und der optimalen Spindelabmessungen*. Schweinfurt: SKF Kugellagerfabriken, 1962.

[97] X. Chen and W. Rowe, “Analysis and simulation of the grinding process. Part I: Generation of the grinding wheel surface,” *International Journal of Machine Tools and Manufacture*, vol. 36, no. 8, pp. 871–882, 1996, doi: 10.1016/0890-6955(96)00116-2.

[98] E. Abele, Y. Altintas, and C. Brecher, “Machine tool spindle units,” *CIRP Annals*, vol. 59, no. 2, pp. 781–802, 2010, doi: 10.1016/j.cirp.2010.05.002.

[99] NSK Americas, *NSK Types.* [Online]. Available: https://www.nskamericas.com/types-2245.htm (accessed: Apr. 11 2023).

[100] A. Fernandez, *Prediction of the bearing damage severity level.* [Online]. Available: https://power-mi.com/content/prediction-bearing-damage-severity-level (accessed: Dec. 18 2020).

[101] Schaeffler Technologies AG & Co. KG, *Wälzlagerpraxis: Handbuch zur Gestaltung und Berechnung von Wälzlagerungen,* 4th ed. Mainz: Vereinigte Fachverl., 2015.

[102] I. Attoui, N. Fergani, N. Boutasseta, B. Oudjani, and A. Deliou, “A new time–frequency method for identification and classification of ball bearing faults,” *Journal of Sound and Vibration*, vol. 397, pp. 241–265, 2017, doi: 10.1016/j.jsv.2017.02.041.

[103] M. Yuwono, Y. Qin, J. Zhou, Y. Guo, B. G. Celler, and S. W. Su, “Automatic bearing fault diagnosis using particle swarm clustering and Hidden Markov Model,” *Engineering Applications of Artificial Intelligence*, vol. 47, pp. 88–100, 2016, doi: 10.1016/j.engappai.2015.03.007.

[104] H. Zhou, J. Chen, G. Dong, and R. Wang, “Detection and diagnosis of bearing faults using shift-invariant dictionary learning and hidden Markov model,” *Mechanical Systems and Signal Processing*, 72-73, pp. 65–79, 2016, doi: 10.1016/j.ymssp.2015.11.022.

[105] J. Cai and Y. Xiao, “Bearing fault diagnosis method based on the generalized S transform time–frequency spectrum de-noised by singular value decomposition,” *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, vol. 233, no. 7, pp. 2467–2477, 2019, doi: 10.1177/0954406218782285.

[106] Y. Li, H. Cao, and X. Chen, “Modelling and vibration analysis of machine tool spindle system with bearing defects,” *IJMMS*, vol. 8, 1/2, p. 33, 2015, doi: 10.1504/IJMMS.2015.071686.

[107] F. Dalvand, M. Kang, S. Dalvand, and M. Pecht, “Detection of Generalized-Roughness and Single-Point Bearing Faults Using Linear Prediction-Based Current Noise Cancellation,” *IEEE Trans. Ind. Electron.*, vol. 65, no. 12, pp. 9728–9738, 2018, doi: 10.1109/TIE.2018.2821645.

[108] P. D. McFadden and J. D. Smith, “The vibration produced by multiple point defects in a rolling element bearing,” *Journal of Sound and Vibration*, vol. 98, no. 2, pp. 263–273, 1985, doi: 10.1016/0022-460X(85)90390-6.

[109] E. Matzan, “Detecting Premature Bearing Failure,” *Machinery Lubrication*, no. 5, 2007. [Online]. Available: https://www.machinerylubrication.com/Read/1041/detecting-bearing-failure

[110] P. Lecinski, “Bearing Problems – Fault Frequency and Artificial Intelligence-Based Methods,” *CBM CONNECT*, 18 Oct., 2021. https://www.cbmconnect.com/bearing-problems-fault-frequency-and-artificial-intelligence-based-methods/ (accessed: Apr. 11 2023).

[111] J. I. Taylor, *The vibration analysis handbook,* 1st ed. Tampa, FL: Vibration Consultants, 1994.

[112] S. Schwendemann and A. Sikora, “Transfer-Learning-Based Estimation of the Remaining Useful Life of Heterogeneous Bearing Types Using Low-Frequency Accelerometers,” *J. Imaging*, vol. 9, no. 2, p. 34, 2023, doi: 10.3390/jimaging9020034.

[113] W. Du, J. Tao, Y. Li, and C. Liu, “Wavelet leaders multifractal features based fault diagnosis of rotating mechanism,” *Mechanical Systems and Signal Processing*, vol. 43, 1-2, pp. 57–75, 2014, doi: 10.1016/j.ymssp.2013.09.003.

[114] Z. Chen, S. Deng, X. Chen, C. Li, R.-V. Sanchez, and H. Qin, “Deep neural networks-based rolling bearing fault diagnosis,” *Microelectronics Reliability*, vol. 75, pp. 327–333, 2017, doi: 10.1016/j.microrel.2017.03.006.

[115] J. P. Patel and S. H. Upadhyay, “Comparison between Artificial Neural Network and Support Vector Method for a Fault Diagnostics in Rolling Element Bearings,” *Procedia Engineering*, vol. 144, pp. 390–397, 2016, doi: 10.1016/j.proeng.2016.05.148.

[116] C. Lu, Z. Wang, and B. Zhou, “Intelligent fault diagnosis of rolling bearing using hierarchical convolutional network based health state classification,” *Advanced Engineering Informatics*, vol. 32, pp. 139–151, 2017, doi: 10.1016/j.aei.2017.02.005.

[117] E. Hering, *Sensoren in Wissenschaft und Technik: Funktionsweise und Einsatzgebiete,* 2nd ed. Wiesbaden: Springer Fachmedien Wiesbaden GmbH, 2018. [Online]. Available: https://ebookcentral.proquest.com/lib/kxp/detail.action?docID=5231087

[118] Balance Systems S.r.l., *B-Safe X System.* [Online]. Available: http://b-safesensor.balancesystems.com/#1573663298445-282073fa-c906 (accessed: Apr. 22 2021).

[119] ifm electronic gmbh, *Systeme zur Schwingungsüberwachung und -diagnose - Beschleunigungssensoren.* [Online]. Available: https://www.ifm.com/de/de/category/070/070_010/070_010_025#!/S/BD/DM/1/D/0/F/0/T/24 (accessed: Apr. 22 2021).

[120] FEMTO ST, *IEEE PHM 2012 Data Challenge.* [Online]. Available: http://web.archive.org/web/20160304041226/http://www.femto-st.fr/f/d/IEEEPHM2012-Challenge-Details.pdf (accessed: Apr. 11 2023).

[121] K. A. Loparo, *Bearing Data Center.* [Online]. Available: http://csegroups.case.edu/bearingdatacenter/home (accessed: Oct. 1 2019).

[122] ifm electronic gmbh 2021, *VSM101 - Accelerometer - ifm electronic.* [Online]. Available: https://www.ifm.com/us/en/product/VSM101 (accessed: Feb. 11 2023).

[123] PCB Piezotronics, *PCB Model 639A91.* [Online]. Available: http://www.pcb.com/products?m=639A91 (accessed: Apr. 11 2023).

[124] *Mechanische Schwingungen - Bewertung der Schwingungen von Maschinen durch Messungen an nicht-rotierenden Teilen - Teil 3: Industrielle Maschinen mit einer Nennleistung über 15 kW und Nenndrehzahlen zwischen 120 min^{-1} und 15000 min^{-1} bei Messungen am Aufstellungsort (ISO 10816-3:2009 + Amd.1:2017)*, 10816-3:2018-01, Deutsches Institut für Normung e.V., Berlin.

[125] X. Ding and Q. He, “Energy-Fluctuated Multiscale Feature Learning With Deep ConvNet for Intelligent Spindle Bearing Fault Diagnosis,” *IEEE Trans. Instrum. Meas.*, vol. 66, no. 8, pp. 1926–1935, 2017, doi: 10.1109/TIM.2017.2674738.

[126] 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, doi: 10.1016/j.cie.2012.02.002.

[127] Y.-K. Hwang, I.-H. Park, K.-S. Paik, and C.-M. Lee, “Development of a variable preload spindle by using an electromagnetic actuator,” *Int. J. Precis. Eng. Manuf.*, vol. 15, no. 2, pp. 201–207, 2014, doi: 10.1007/s12541-014-0326-9.

[128] M. Engeler, A. Elmiger, A. Kunz, D. Zogg, and K. Wegener, “Online Condition Monitoring Tool for Automated Machinery,” *Procedia CIRP*, vol. 58, pp. 323–328, 2017, doi: 10.1016/j.procir.2017.04.003.

[129] M. Cerrada *et al.,* “A review on data-driven fault severity assessment in rolling bearings,” *Mechanical Systems and Signal Processing*, vol. 99, pp. 169–196, 2018, doi: 10.1016/j.ymssp.2017.06.012.

[130] P. Junge, “ifm Programmierhandbuch octavis VES004 V1.20.11,” IFM, Jun. 2017.

[131] S. Zhang, S. Zhang, B. Wang, and T. G. Habetler, “Deep Learning Algorithms for Bearing Fault Diagnostics—A Comprehensive Review,” *IEEE Access*, vol. 8, pp. 29857–29881, 2020, doi: 10.1109/ACCESS.2020.2972859.

[132] H. Liu, Z. Mo, H. Zhang, X. Zeng, J. Wang, and Q. Miao, “Investigation on Rolling Bearing Remaining Useful Life Prediction: A Review,” in *2018 Prognostics and System Health Management Conference (PHM-Chongqing)*, Chongqing, Oct. 2018 - Oct. 2018, pp. 979–984.

[133] S. Schwendemann, Z. Amjad, and A. Sikora, “Bearing fault diagnosis with intermediate domain based Layered Maximum Mean Discrepancy: A new transfer learning approach,” *Engineering Applications of Artificial Intelligence*, vol. 105, 104415, 2021, doi: 10.1016/j.engappai.2021.104415.

[134] Y. Li, M. Xu, H. Zhao, and W. Huang, “Hierarchical fuzzy entropy and improved support vector machine based binary tree approach for rolling bearing fault diagnosis,” *Mechanism and Machine Theory*, vol. 98, pp. 114–132, 2016, doi: 10.1016/j.mechmachtheory.2015.11.010.

[135] Y. Wang, S. Kang, Y. Jiang, G. Yang, L. Song, and V. I. Mikulovich, “Classification of fault location and the degree of performance degradation of a rolling bearing based on an improved hyper-sphere-structured multi-class support vector machine,” *Mechanical Systems and Signal Processing*, vol. 29, pp. 404–414, 2012, doi: 10.1016/j.ymssp.2011.11.015.

[136] W. Zhang, G. Peng, C. Li, Y. Chen, and Z. Zhang, “A New Deep Learning Model for Fault Diagnosis with Good Anti-Noise and Domain Adaptation Ability on Raw Vibration Signals,” *Sensors (Basel, Switzerland)*, vol. 17, no. 2, pp. 425–446, 2017, doi: 10.3390/s17020425.

[137] X. Li, W. Zhang, Q. Ding, and J.-Q. Sun, “Multi-Layer domain adaptation method for rolling bearing fault diagnosis,” *Signal Processing*, vol. 157, pp. 180–197, 2019, doi: 10.1016/j.sigpro.2018.12.005.

[138] X. Li, W. Zhang, and Q. Ding, “Cross-Domain Fault Diagnosis of Rolling Element Bearings Using Deep Generative Neural Networks,” *IEEE Trans. Ind. Electron.*, pp. 5525–5534, 2018, doi: 10.1109/TIE.2018.2868023.

[139] T. Han, C. Liu, W. Yang, and D. Jiang, "Deep transfer network with joint distribution adaptation: A new intelligent fault diagnosis framework for industry application," *ISA transactions*, vol. 97, pp. 269–281, 2020, doi: 10.1016/j.isatra.2019.08.012.

[140] T. Ince, S. Kiranyaz, L. Eren, M. Askar, and M. Gabbouj, "Real-Time Motor Fault Detection by 1-D Convolutional Neural Networks," *IEEE Trans. Ind. Electron.*, vol. 63, no. 11, pp. 7067–7075, 2016, doi: 10.1109/TIE.2016.2582729.

[141] D. Verstraete, A. Ferrada, E. L. Droguett, V. Meruane, and M. Modarres, "Deep Learning Enabled Fault Diagnosis Using Time-Frequency Image Analysis of Rolling Element Bearings," *Shock and Vibration*, vol. 2017, pp. 1–17, 2017, doi: 10.1155/2017/5067651.

[142] A. Kızrak, *Comparison of Activation Functions for Deep Neural Networks.* [Online]. Available: https://towardsdatascience.com/comparison-of-activation-functions-for-deep-neural-networks-706ac4284c8a (accessed: May 5 2021).

[143] L. Jing, M. Zhao, P. Li, and X. Xu, "A convolutional neural network based feature learning and fault diagnosis method for the condition monitoring of gearbox," *Measurement*, vol. 111, pp. 1–10, 2017, doi: 10.1016/j.measurement.2017.07.017.

[144] W. You, C.-Q. Shen, and Z.-K. Zhu, "Bearing Fault Diagnosis Using Convolution Neural Network and Support Vector Regression," in *International Conference on Mechanical Engineering and Control Automation (ICMECA 2017)*, Lancaster, Pennsylvania 17602 U.S.A.: DEStech Publications, Inc., 2017.

[145] L. Schmarje, M. Santarossa, S.-M. Schroder, and R. Koch, "A Survey on Semi-, Self- and Unsupervised Learning for Image Classification," *IEEE Access*, vol. 9, pp. 82146–82168, 2021, doi: 10.1109/ACCESS.2021.3084358.

[146] B. Yang, Y. Lei, F. Jia, and S. Xing, "An intelligent fault diagnosis approach based on transfer learning from laboratory bearings to locomotive bearings," *Mechanical Systems and Signal Processing*, vol. 122, pp. 692–706, 2019, doi: 10.1016/j.ymssp.2018.12.051.

[147] H. Zhiyi, S. Haidong, J. Lin, C. Junsheng, and Y. Yu, "Transfer fault diagnosis of bearing installed in different machines using enhanced deep auto-encoder," *Measurement*, vol. 152, p. 107393, 2020, doi: 10.1016/j.measurement.2019.107393.

[148] L. Guo, N. Li, F. Jia, Y. Lei, and J. Lin, "A recurrent neural network based health indicator for remaining useful life prediction of bearings," *Neurocomputing*, vol. 240, pp. 98–109, 2017, doi: 10.1016/j.neucom.2017.02.045.

[149] A. Soualhi, K. Medjaher, and N. Zerhouni, "Bearing Health Monitoring Based on Hilbert–Huang Transform, Support Vector Machine, and Regression," *IEEE Trans. Instrum. Meas.*, vol. 64, no. 1, pp. 52–62, 2015, doi: 10.1109/TIM.2014.2330494.

[150] T. Liu, J. Chen, and G. Dong, “Zero crossing and coupled hidden Markov model for a rolling bearing performance degradation assessment,” *Journal of Vibration and Control*, vol. 20, no. 16, pp. 2487–2500, 2014, doi: 10.1177/1077546313479992.

[151] A. Soylemezoglu, S. Jagannathan, and C. Saygin, “Mahalanobis Taguchi System (MTS) as a Prognostics Tool for Rolling Element Bearing Failures,” *J. Manuf. Sci. Eng.*, vol. 132, no. 5, 51014, 2010, doi: 10.1115/1.4002545.

[152] J. Ben Ali, B. Chebel-Morello, L. Saidi, S. Malinowski, and F. Fnaiech, “Accurate bearing remaining useful life prediction based on Weibull distribution and artificial neural network,” *Mechanical Systems and Signal Processing*, 56-57, pp. 150–172, 2015, doi: 10.1016/j.ymssp.2014.10.014.

[153] Z. Huang, Z. Xu, X. Ke, W. Wang, and Y. Sun, “Remaining useful life prediction for an adaptive skew-Wiener process model,” *Mechanical Systems and Signal Processing*, vol. 87, pp. 294–306, 2017, doi: 10.1016/j.ymssp.2016.10.027.

[154] Z. Liu, M. J. Zuo, and Y. Qin, “Remaining useful life prediction of rolling element bearings based on health state assessment,” *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, vol. 230, no. 2, pp. 314–330, 2016, doi: 10.1177/0954406215590167.

[155] E. Sturisno, H. Oh, A. S. S. Vasan, and M. Pecht, *IEEE Conference on Prognostics and Health Management (PHM), 2012: 18 - 21 June 2012, Denver, Colorado*. Piscataway, NJ: IEEE, 2012. [Online]. Available: http://ieeexplore.ieee.org/servlet/opac?punumber=6294524

[156] A. Malhi, R. Yan, and R. X. Gao, “Prognosis of Defect Propagation Based on Recurrent Neural Networks,” *IEEE Transactions on Instrumentation and Measurement*, vol. 60, pp. 703–711, 2011, doi: 10.1109/TIM.2010.2078296.

[157] G. Zhang, W. Liang, B. She, and F. Tian, “Rotating Machinery Remaining Useful Life Prediction Scheme Using Deep-Learning-Based Health Indicator and a New RVM,” *Shock and Vibration*, vol. 2021, pp. 1–14, 2021, doi: 10.1155/2021/8815241.

[158] P. Xia, Y. Huang, P. Li, C. Liu, and L. Shi, “Fault Knowledge Transfer Assisted Ensemble Method for Remaining Useful Life Prediction,” *IEEE Trans. Ind. Inf.*, vol. 18, no. 3, pp. 1758–1769, 2022, doi: 10.1109/TII.2021.3081595.

[159] G. Huang, Y. Zhang, and J. Ou, “Transfer remaining useful life estimation of bearing using depth-wise separable convolution recurrent network,” *Measurement*, vol. 176, p. 109090, 2021, doi: 10.1016/j.measurement.2021.109090.

[160] B. Zhang, S. Zhang, and W. Li, “Bearing performance degradation assessment using long short-term memory recurrent network,” *Computers in Industry*, vol. 106, pp. 14–29, 2019, doi: 10.1016/j.compind.2018.12.016.

[161] C. Knieke, M. Mansouri, and G. Telleschi, Eds., *ICONS 2020: The Fifteenth International Conference on Systems : February 23-27, 2020, Lisbon, Portugal*. Wilmington, DE, USA: IARIA, 2020. [Online]. Available: http://thinkmind.org/download_full.php?instance=ICONS+2020

[162] L. Sadouk, “CNN Approaches for Time Series Classification,” in *Time Series Analysis - Data, Methods, and Applications*, C.-K. Ngan, Ed.: IntechOpen, 2019.

[163] L. Zhang, S. Wang, G.-B. Huang, W. Zuo, J. Yang, and D. Zhang, “Manifold Criterion Guided Transfer Learning via Intermediate Domain Generation,” Mar. 2019. [Online]. Available: http://arxiv.org/pdf/1903.10211v1

[164] X. Zhang, Y. Liang, J. Zhou, and Y. zang, “A novel bearing fault diagnosis model integrated permutation entropy, ensemble empirical mode decomposition and optimized SVM,” *Measurement*, vol. 69, pp. 164–179, 2015, doi: 10.1016/j.measurement.2015.03.017.

[165] G. Fan, J. Li, and H. Hao, “Vibration signal denoising for structural health monitoring by residual convolutional neural networks,” *Measurement*, vol. 157, p. 107651, 2020, doi: 10.1016/j.measurement.2020.107651.

[166] Q. He, X. Wang, and Q. Zhou, “Vibration sensor data denoising using a time-frequency manifold for machinery fault diagnosis,” *Sensors (Basel, Switzerland)*, vol. 14, no. 1, pp. 382–402, 2013, doi: 10.3390/s140100382.

[167] S. Braun, “The synchronous (time domain) average revisited,” *Mechanical Systems and Signal Processing*, vol. 25, no. 4, pp. 1087–1102, 2011, doi: 10.1016/j.ymssp.2010.07.016.

[168] W. He, Q. Miao, M. Azarian, and M. Pecht, “Health monitoring of cooling fan bearings based on wavelet filter,” *Mechanical Systems and Signal Processing*, 64-65, pp. 149–161, 2015, doi: 10.1016/j.ymssp.2015.04.002.

[169] B. Dolenc, P. Boškoski, and Đ. Juričić, “Distributed bearing fault diagnosis based on vibration analysis,” *Mechanical Systems and Signal Processing*, 66-67, pp. 521–532, 2016, doi: 10.1016/j.ymssp.2015.06.007.

[170] N. Cui, “Applying Gradient Descent in Convolutional Neural Networks,” *J. Phys.: Conf. Ser.*, vol. 1004, p. 12027, 2018, doi: 10.1088/1742-6596/1004/1/012027.

[171] J. W. Tan, *CS231n: Convolutional Neural Networks for Visual Recognition.* [Online]. Available: https://cs231n.github.io/convolutional-networks/ (accessed: Apr. 22 2021).

[172] K. Vernekar, H. Kumar, and K. V. Gangadharan, “Gear Fault Detection Using Vibration Analysis and Continuous Wavelet Transform,” *Procedia Materials Science*, vol. 5, pp. 1846–1852, 2014, doi: 10.1016/j.mspro.2014.07.492.

[173] V. Kavana and M. Neethi, “Fault Analysis and Predictive Maintenance of Induction Motor Using Machine Learning,” in *2018 International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT)*, Msyuru, India, 2018, pp. 963–966.

[174] J. Li, W. Wu, and Di Xue, “Research on transfer learning algorithm based on support vector machine,” *IFS*, vol. 38, no. 4, pp. 4091–4106, 2020, doi: 10.3233/JIFS-190055.

[175] S. Pandey, *How to choose the size of the convolution filter or Kernel size for CNN?* [Online]. Available: https://medium.com/analytics-vidhya/how-to-choose-the-size-of-the-convolution-filter-or-kernel-size-for-cnn-86a55a1e2d15 (accessed: Apr. 20 2021).

[176] J. Brownlee, *How to Choose Loss Functions When Training Deep Learning Neural Networks.* [Online]. Available: https://machinelearningmastery.com/how-to-choose-loss-functions-when-training-deep-learning-neural-networks/ (accessed: Apr. 22 2021).

[177] A. Aniculaesei, A. Vorwald, M. Zhang, and A. Rausch, “Architecture-based Hybrid Approach to Verify Safety-critical Automotive System Functions by Combining Data-driven and Formal Methods,” in *2021 IEEE 18th International Conference on Software Architecture Companion (ICSA-C)*, Stuttgart, Germany, 2021, pp. 1–10.

[178] “IEEE PHM 2012 Prognostic challenge: Outline, Experiments, Scoring of results, Winners,” 2012.

[179] Z.-H. Liu *et al.,* “A Regularized LSTM Method for Predicting Remaining Useful Life of Rolling Bearings,” *Int. J. Autom. Comput.*, vol. 18, no. 4, pp. 581–593, 2021, doi: 10.1007/s11633-020-1276-6.

[180] S. Yao, Q. Kang, M. Zhou, M. J. Rawa, and A. Abusorrah, “A survey of transfer learning for machinery diagnostics and prognostics,” *Artif Intell Rev*, 2022, doi: 10.1007/s10462-022-10230-4.

[181] H. Naeem and A. A. Bin-Salem, “A CNN-LSTM network with multi-level feature extraction-based approach for automated detection of coronavirus from CT scan and X-ray images,” *Applied Soft Computing*, vol. 113, p. 107918, 2021, doi: 10.1016/j.asoc.2021.107918.

[182] Pin Lim, Chi Keong Goh, Kay Chen Tan, and Partha Dutta, “Estimation of Remaining Useful Life Based on Switching Kalman Filter Neural Network Ensemble,” *PHM_CONF*, vol. 6, no. 1, 2014, doi: 10.36001/phmconf.2014.v6i1.2348.

[183] Y. Mo, Q. Wu, X. Li, and B. Huang, “Remaining useful life estimation via transformer encoder enhanced by a gated convolutional unit,” *J Intell Manuf*, vol. 32, no. 7, pp. 1997–2006, 2021, doi: 10.1007/s10845-021-01750-x.

[184] M. M. Khan, P. W. Tse, and J. Yang, “A Novel Framework for Online Remaining Useful Life Prediction of an Industrial Slurry Pump,” *Applied Sciences*, vol. 12, no. 10, p. 4839, 2022, doi: 10.3390/app12104839.

[185] S.-H. Noh, “Analysis of Gradient Vanishing of RNNs and Performance Comparison,” *Information*, vol. 12, no. 11, p. 442, 2021, doi: 10.3390/info12110442.

[186] X. Song, D. Zhu, P. Liang, and L. An, “A new bearing fault diagnosis method using elastic net transfer learning and LSTM,” *IFS*, vol. 40, no. 6, pp. 12361–12369, 2021, doi: 10.3233/JIFS-210503.

[187] M. Jalayer, C. Orsenigo, and C. Vercellis, “Fault detection and diagnosis for rotating machinery: A model based on convolutional LSTM, Fast Fourier and continuous wavelet transforms,” *Computers in Industry*, vol. 125, p. 103378, 2021, doi: 10.1016/j.compind.2020.103378.

[188] K. Patra, A. K. Jha, T. Szalay, J. Ranjan, and L. Monostori, “Artificial neural network based tool condition monitoring in micro mechanical peck drilling using thrust force signals,” *Precision Engineering*, vol. 48, pp. 279–291, 2017, doi: 10.1016/j.precisioneng.2016.12.011.

[189] D. M. D’Addona, D. Matarazzo, P. R. de Aguiar, E. C. Bianchi, and C. H. Martins, “Neural Networks Tool Condition Monitoring in Single-point Dressing Operations,” *Procedia CIRP*, vol. 41, pp. 431–436, 2016, doi: 10.1016/j.procir.2016.01.001.

[190] A. Ramdas, S. J. Reddi, B. Poczos, A. Singh, and L. Wasserman, “Adaptivity and Computation-Statistics Tradeoffs for Kernel and Distance based High Dimensional Two Sample Testing,” Aug. 2015. [Online]. Available: http://arxiv.org/pdf/1508.00655v1

[191] D. J. Sutherland *et al.,* “Generative Models and Model Criticism via Optimized Maximum Mean Discrepancy,” Nov. 2016. [Online]. Available: http://arxiv.org/pdf/1611.04488v6

[192] S. Hossain, F. T. Johora, J. P. Müller, S. Hartmann, and A. Reinhardt, “SFMGNet: A Physics-based Neural Network To Predict Pedestrian Trajectories,” Feb. 2022. [Online]. Available: http://arxiv.org/pdf/2202.02791v1

[193] S. Porotsky and Z. Bluvband, “Remaining useful life estimation for systems with non-trendability behaviour,” in *2012 IEEE Conference on Prognostics and Health Management*, Denver, CO, USA, 062012, pp. 1–6.

[194] Y. Zheng, “Predicting Remaining Useful Life Based on Hilbert–Huang Entropy with Degradation Model,” *Journal of Electrical and Computer Engineering*, vol. 2019, pp. 1–11, 2019, doi: 10.1155/2019/3203959.

[195] Z. Xu, Y. Guo, and J. H. Saleh, “Remaining useful life prediction with uncertainty quantification: development of a highly accurate model for rotating machinery,” Apr. 2023. [Online]. Available: https://arxiv.org/pdf/2109.11579

[196] Y. Yoo and J.-G. Baek, “A Novel Image Feature for the Remaining Useful Lifetime Prediction of Bearings Based on Continuous Wavelet Transform and Convolutional Neural Network,” *Applied Sciences*, vol. 8, no. 7, p. 1102, 2018, doi: 10.3390/app8071102.

A Appendix

A.1 General

A.1.1 Assignment of Datasets

For each training and test run, the datasets are assigned to training and test data in the same way. The original assignment was random, but this assignment is kept constant according to Table 29 for each verification step.

Table 29: Assignment of the different datasets for test and training data.

Dataset	Run 1		Run 2	
	Training data	Test data	Training data	Test data
CWRU	130, 133, 144, 146, 147, 156, 159, 160, 197, 198, 199, 200, 234, 235, 237, 247, 248, 259, 260, 261, 105, 108, 169, 170, 172, 209, 210, 211, 3001, 3002, 3003, 3004, 97, 98, 99	249, 158, 145, 132, 236, 246, 131, 258, 171, 212, 107, 106, 100	100, 97, 98, 130, 132, 133, 144, 146, 147, 159, 160, 197, 198, 199, 200, 234, 236, 237, 246, 247, 248, 249, 258, 105, 106, 107, 108, 169, 170, 171, 172, 210, 212, 3001, 3002	99, 131, 235, 156, 158, 260, 261, 145, 259, 211, 209, 3003, 3004
Junker	5000000808_c10_Y, 5000000808_c50_X, 5000000808_c50_Y, 5000000808_w10_X, 5000000808_w10_Y, 5000000808_w50_X, 5000000808_w50_Y, 5000000808_w100_X, 5000003177_c10_Y, 5000003177_c50_X, 5000003177_c50_Y, 5000003177_w10_X, 5000003177_w10_Y, 5000003177_w50_X, 5000003177_w50_Y, 5000003177_w100_Y, 5000003796_c10_X, 5000003796_c50_Y, 5000003796_w10_Y, 5000003796_w100_Y, 060412019_c10_X, 060412019_w10_X, 060412019_w10_Y, 060412019_w100_Y, 060706012_c10_X, 060706012_c10_Y, 060706012_w10_X, 060706012_w100_X, 060302049_w50_Y, 060302049_w100_X, 060302049_w100_Y,	5000003796_w10_X, 5000003177_c10_X, 5000003796_c50_X, 5000003796_w100_X, 5000000808_w100_Y, 5000003177_w100_X, 5000000808_c10_X, 5000003796_c10_Y, 060706012_w10_Y, 060412019_c10_Y, 060706012_w100_Y, 060302049_w50_X, 060707037_w10_X	060302049_w50_Y, 060302049_w100_Y, 060707037_w10_X, 060707037_w50_X, 060707037_w100_X, 060412019_c50_X, 060412019_c50_Y, 5000000808_c10_X, 5000000808_c10_Y, 5000000808_c50_X, 5000000808_c50_Y, 5000000808_w10_X, 5000000808_w50_X, 5000000808_w50_Y, 5000000808_w100_Y, 5000003177_c10_Y, 5000003177_c50_X, 5000003177_w10_X, 5000003177_w10_Y, 5000003177_w50_X, 5000003177_w50_Y, 5000003177_w100_X, 5000003177_w100_Y, 5000003796_c50_X, 5000003796_c50_Y, 5000003796_w10_Y, 5000003796_w100_X, 060412019_c10_X, 060412019_c10_Y, 060412019_w10_X, 060412019_w100_Y,	060302049_w50_X, 060302049_w100_X, 5000000808_w100_X, 5000003177_c10_X, 5000003796_w100_Y, 5000003796_c10_X, 5000000808_w10_Y, 5000003796_w10_X, 5000003796_c10_Y, 5000003177_c50_Y, 060706012_w100_Y, 060412019_w10_Y, 060706012_w10_Y

	060707037_w50_X, 060707037_w100_X, 060412019_c50_X, 060412019_c50_Y		060706012_c10_X, 060706012_c10_Y, 060706012_w10_X, 060706012_w100_X	

A.2 Verification of the Intermediate Domain

A.2.1 Frequency Conversion Methods

This test scenario is used to benchmark different frequency conversion methods. The test dataset is the drive-end dataset of the CWRU. The images for the input are created based on 0.2 second time slices and an image size of 64x64 pixels. The compared signal processing techniques are envelope, windowed envelope, HHT, and S-transform. Since the envelope is a frequency-domain technique, the resulting image is only 64x1 pixel in size. Therefore, a 1D CNN is used for the envelope, whereas a 2D CNN is used for all other techniques. The results, presented in Figure 62, show that the windowed envelope is the best technique in terms of classification accuracy.

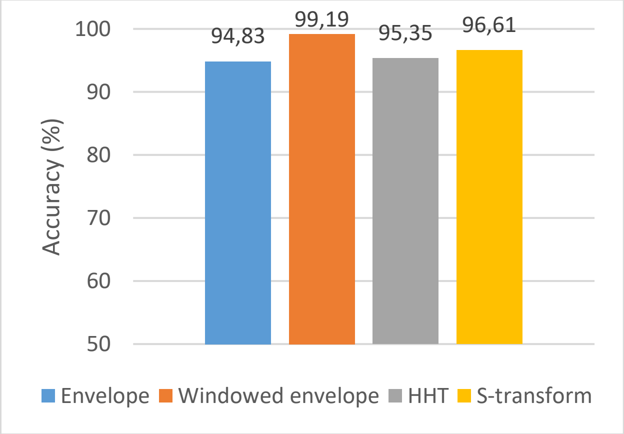

Figure 62: Classification accuracies of three different labels for bearings with different frequency and time-frequency conversions. The best accuracy is achieved with the windowed envelope.

A.2.2 Signal Segmentation

Sensors used in machines provide a stream of measuring values. These must be segmented before their usage. This test scenario shows the influence of the segment length. During this process, an optimum length is also determined. Therefore, the drive-end dataset provided by the CWRU was used. The images for the input are created based on 0.11, 0.2, and 0.7 second time slices and an image size of 64x64 pixels. The raw signal data of these slices was converted with the help of the windowed envelope technique. As shown in Figure 63, all accuracies are nearly equal.

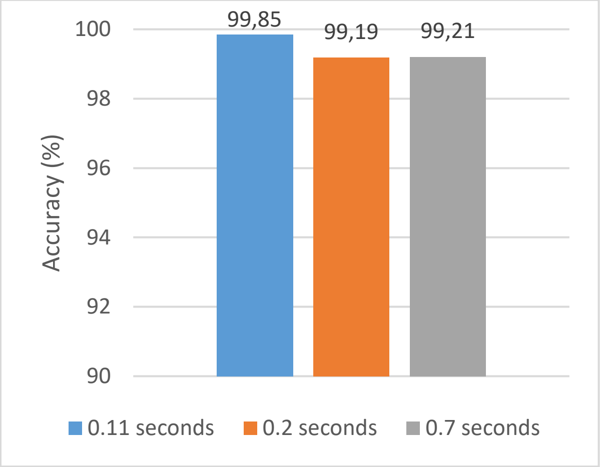

Figure 63: Accuracies of samples of different lengths. All samples are generated with the windowed envelope method. All accuracies are between 99% and 100%.

A.2.3 Different Bandpass Width

When selecting a frequency-selective filter, it is essential to select the bandpass width. As stated in Section 4.5, this must be done to respect the wear-out of components and manufacturing tolerances. Therefore, a test scenario with a bearing dataset of a real-world scenario is used. This dataset is a bearing dataset of a grinding spindle with noise. The bearing fault frequencies are filtered with a bandpass width of 5 Hz, 10 Hz, and 20 Hz. For each fault frequency, the first four harmonics are used. As shown in Figure 64, the best accuracy is reached with a bandpass width of 10 Hz.

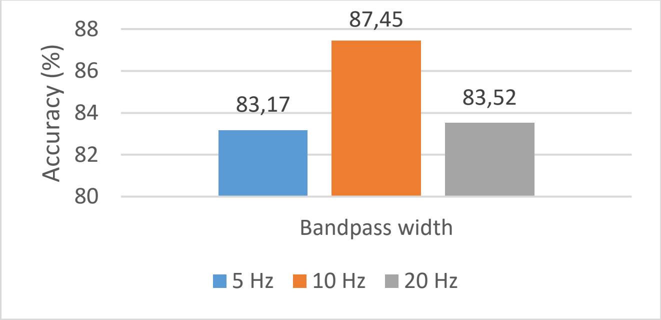

Figure 64: Accuracies for different bandpass widths of a frequency-selective filter, which uses four harmonics. The best accuracy is achieved with a width of 10 Hz. The lower accuracy for 5 Hz might be related to the fact that the actual fault frequency in some samples defers more than 5 Hz through wear-out and manufacturing tolerances. On the other hand, the 20 Hz frequency band might be too wide, so that noise gets into the band, which can actually be filtered out.

A.2.4 Frequency-Selective Filter with Different Harmonics

The proposed intermediate domain is based on using the harmonics of fault frequencies. This test scenario is employed to illustrate the importance of the harmonics. Therefore, the same dataset as in A.2.3 is used. All images are based on 0.2 second time slices and an image size of 64x64 pixels. A frequency band of 10 Hz is used for the frequency-selective filter. As shown in Figure 65, the highest accuracy is reached by using four harmonics. The results also show that if the number of harmonics is too low, an information loss exists that leads to a lower classification accuracy than using the entire windowed envelope.

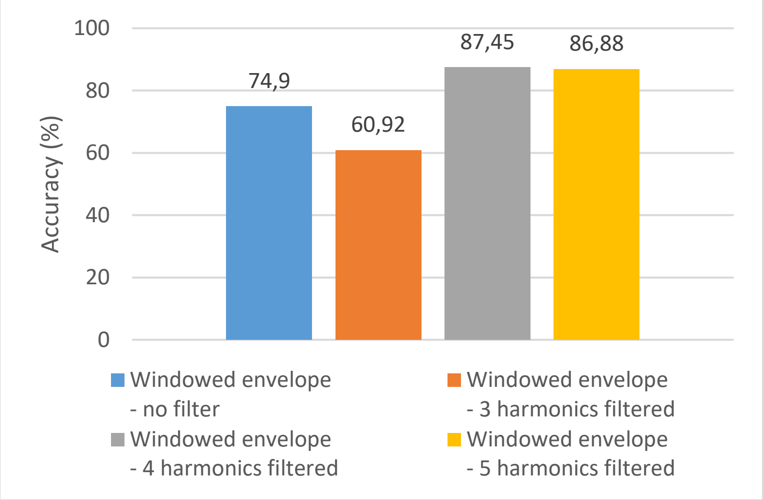

Figure 65: Accuracies for bearing fault classification with the help of a windows envelope in a noisy environment. The highest accuracy can be reached by applying a frequency-selective filter with four harmonics.

A.2.5 Image Size

For evaluating the effect of the size of the input images, the dataset of the CWRU dataset and of the spindle dataset have been used. Images in a 64x64 pixel and 128x128-pixel resolution were created from these datasets. As shown in Figure 66, the accuracies of both resolutions are nearly equal. In one case, the 64x64 pixel image has a slightly better result; in the other, the 128x128 pixel image has a better result. On average, the accuracy of the 64x64-pixel image is better than that of the 128x128-pixel image.

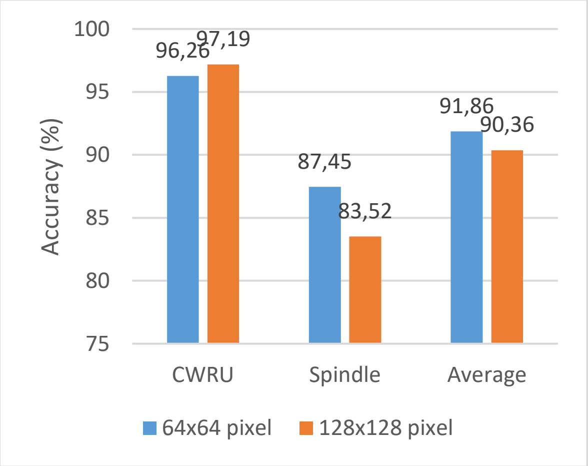

Figure 66: Accuracies of different sizes of the input image. For the CWRU dataset, a 128x128 pixel image has a slightly better accuracy. For the Spindle dataset, the 64x64 performs better. On average, the 64x64 pixel has the best accuracy.

A.3 Verifications of the Transfer Learning Approach for Classification

In order to validate the decisions of the different development steps for the presented classification approach of Chapter 5, different test scenarios have been used. If not mentioned otherwise, the test scenarios use the classification approach developed in this thesis. Bearing datasets, which are split into 70% training data and 30% test data, are used for all scenarios. Each test scenario is run two times with a different training and test data split. The mean of the accuracies of both runs is used as the result. The split is identical between all test cases in one test scenario.

A.3.1 Number of Convolutional Layers

This test scenario examines the influence of the number of convolutional layers in a row before a pooling layer appears in a CNN model. The model of the CNN is always the same as that described in Section 5.4.2. The only difference is the number of convolutional layers (single, double, and triple). For the verification, samples of the CWRU dataset of the drive-end side and a spindle dataset were used. As shown in Figure 67, the best accuracy is reached when using a double convolutional layer approach in both cases.

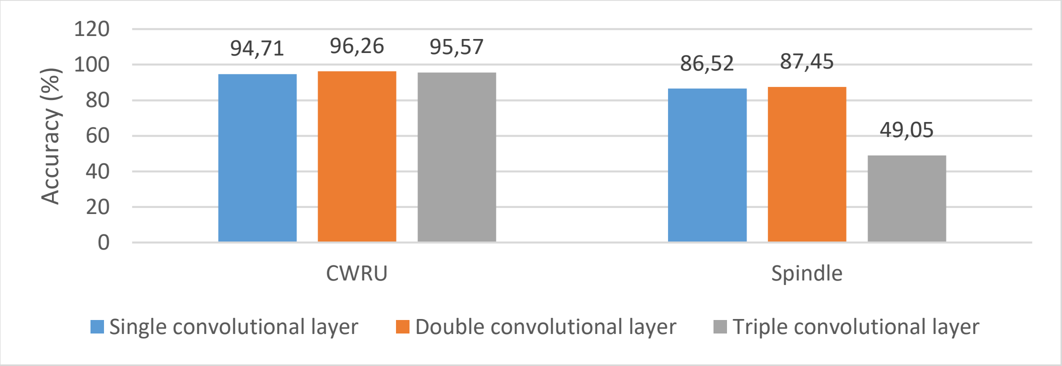

Figure 67: Comparison of a different number of convolutional layers in series. Two different test cases have been used. For both tested datasets, the double layer approach has the best accuracy.

A.3.2 Dropout Factor

This test scenario examines different values of the dropout factor during the training of a CNN. For the verification, samples of the CWRU dataset on the drive-end side were used. As shown in Figure 68, the accuracies seem to be independent of the dropout factor. There is only a small difference, which might be caused by the stochastic nature of the algorithms and the numerical precision.

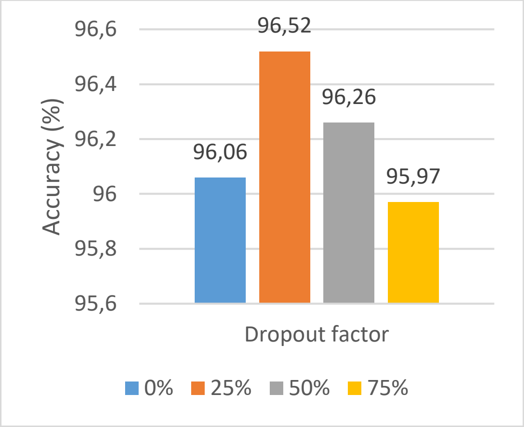

Figure 68: Different values for the dropout factor in the fully connected layers of the CNN. There is no significant difference between the accuracies.

A.3.3 Loss function

This test scenario compares the accuracy of the Kullback Leibler divergence with the cross-entropy. Therefore, a spindle dataset and the dataset of the CWRU are used. The samples are supervised trained, one time with the Kullback Leibler divergence as loss function and the second time with the cross-entropy as loss function. As can be seen in Figure 69, none of them is superior. In one case, cross-entropy as a loss function has a slightly better accuracy. In the other case, Kullback Leibler divergence has a better accuracy.

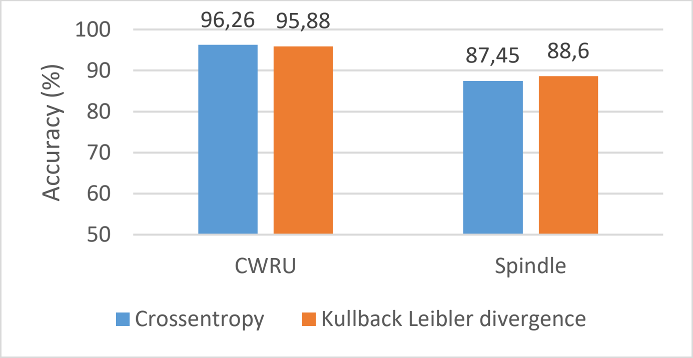

Figure 69: Resulting accuracies of CNN trainings with cross-entropy and Kullback Leiber divergence. None of them is superior.

A.3.4 MMD as Basis for LMMD

To ensure that MMD is a well-suited starting point for the LMMD technique, a comparison of the LMMD approach with other loss functions was performed. For this purpose, the University of Paderborn dataset was used as the source dataset. The target dataset was a spindle dataset with different rotational speeds. The training scenario was semi-supervised. MMD, MK-MMD, and CORAL were tested. MMD and CORAL performed almost equally well.

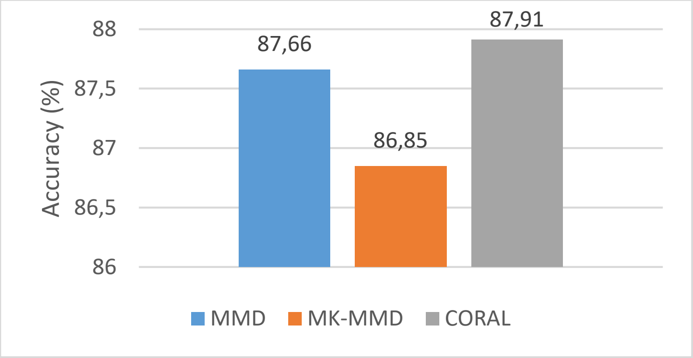

Figure 70: Comparing the results of using loss function as the basis for the LMMD approach. The best results can be achieved with CORAL and MMD.

A.4 Benchmark of the Transfer Learning Approach for Classification

This chapter shows the confusion matrices of the benchmark of Section 7.3. This benchmark is based on two different transfer-learning tasks. For each task, five runs with a random dataset assignment to training and test data have been performed. The first task was to transfer knowledge from drive-end bearings to fan-end bearings. The confusion matrices of these five runs are presented in Figure 71 through Figure 75. The second task was to transfer knowledge from fan-end bearings to drive-end bearings. For this, the results are shown in Figure 71 through Figure 80.

<u>Drive-End to Fan-End:</u>

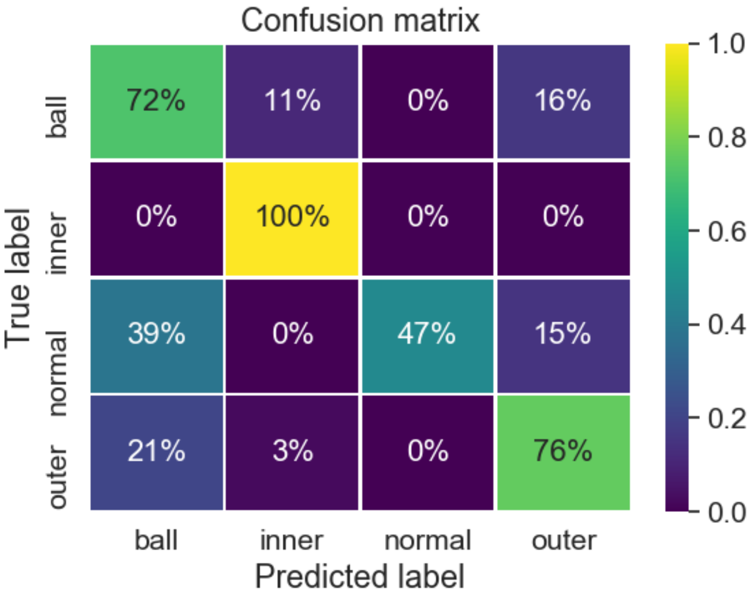

Figure 71: Confusion matrix for the transfer-learning task from drive-end to fan-end. This matrix shows run 1 with an accuracy of 73.82%.

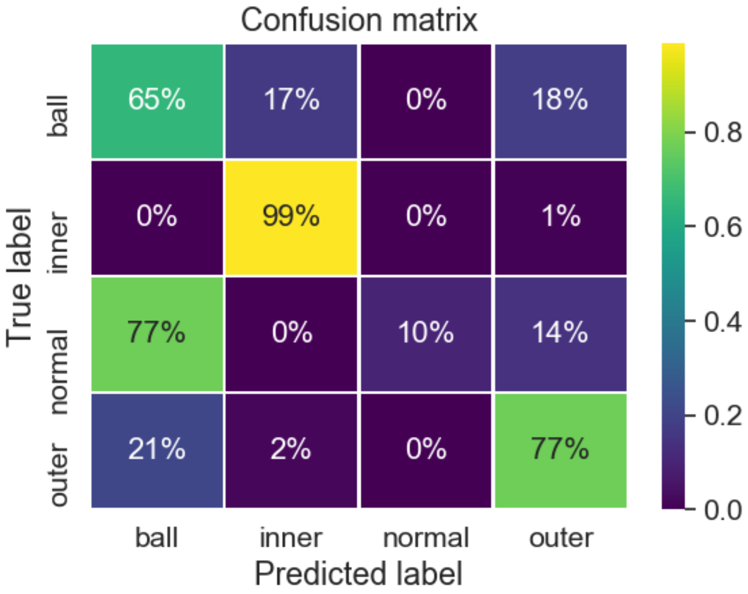

Figure 72: Confusion matrix for the transfer-learning task from drive-end to fan-end. This matrix shows run 2 with an accuracy of 62.78%.

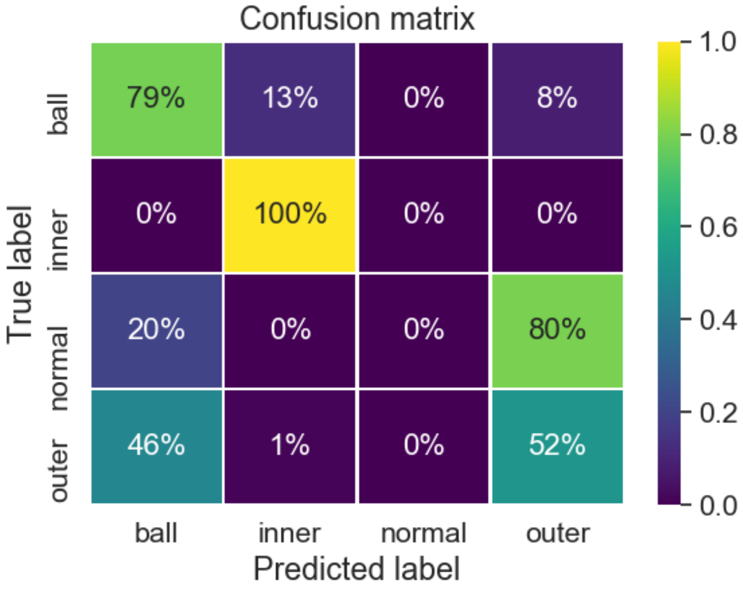

Figure 73: Confusion matrix for the transfer-learning task from drive-end to fan- end. This matrix shows run 3 with an accuracy of 58.03%.

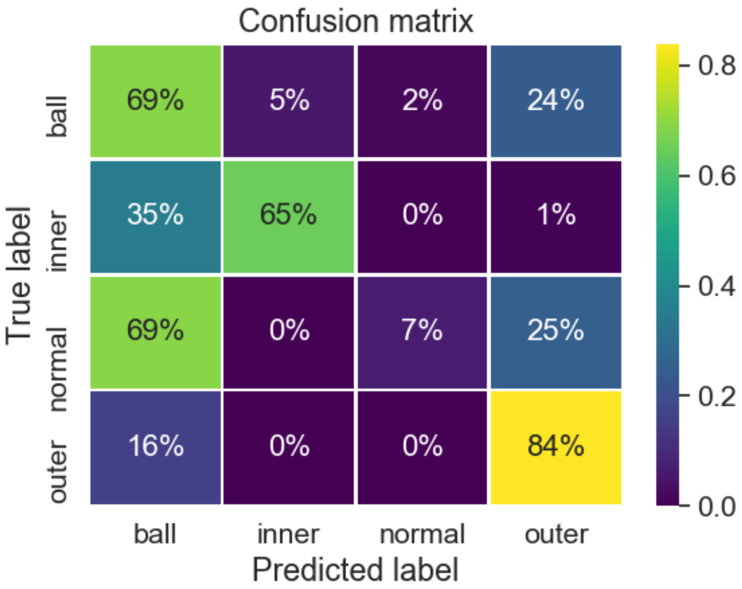

Figure 74: Confusion matrix for the transfer-learning task from drive-end to fan-end. This matrix shows run 4 with an accuracy of 56.05%.

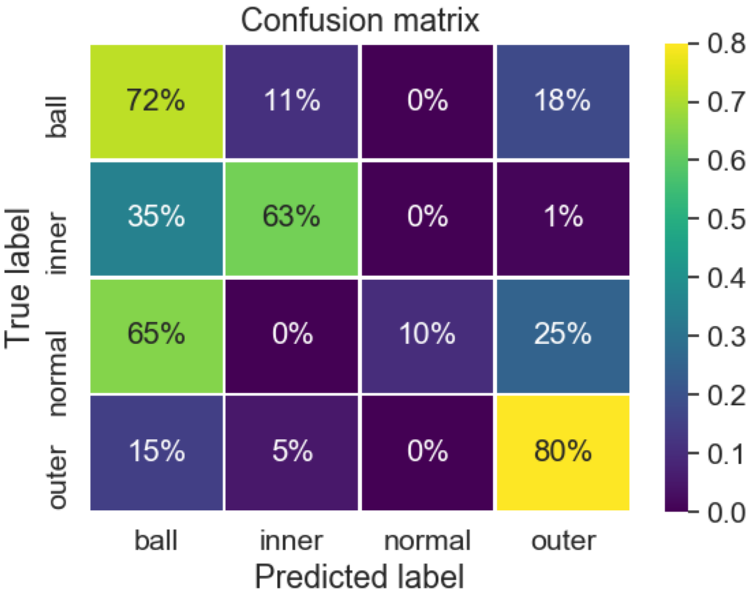

Figure 75: Confusion matrix for the transfer-learning task from drive-end to fan-end. This matrix shows run 5 with an accuracy of 56.18%.

<u>Fan-End to Drive-End:</u>

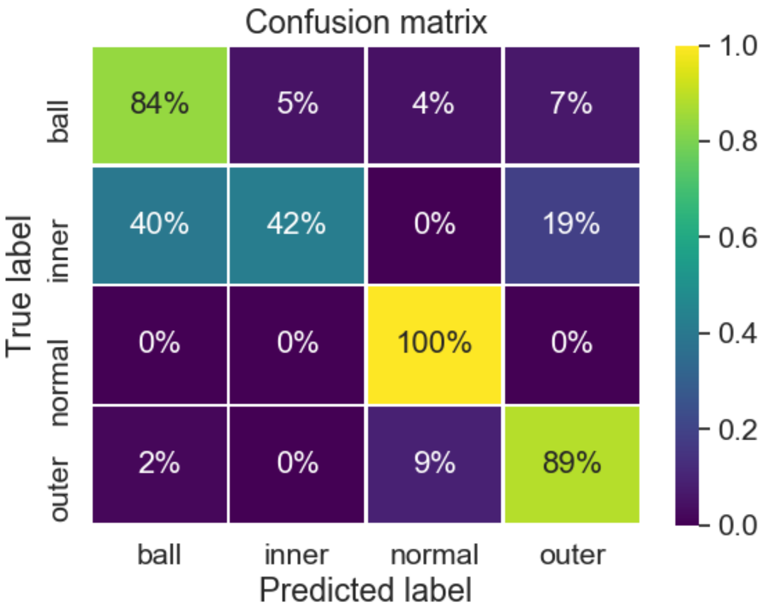

Figure 76: Confusion matrix for the transfer-learning task from fan-end to drive-end. This matrix shows run 1 with an accuracy of 78.70%.

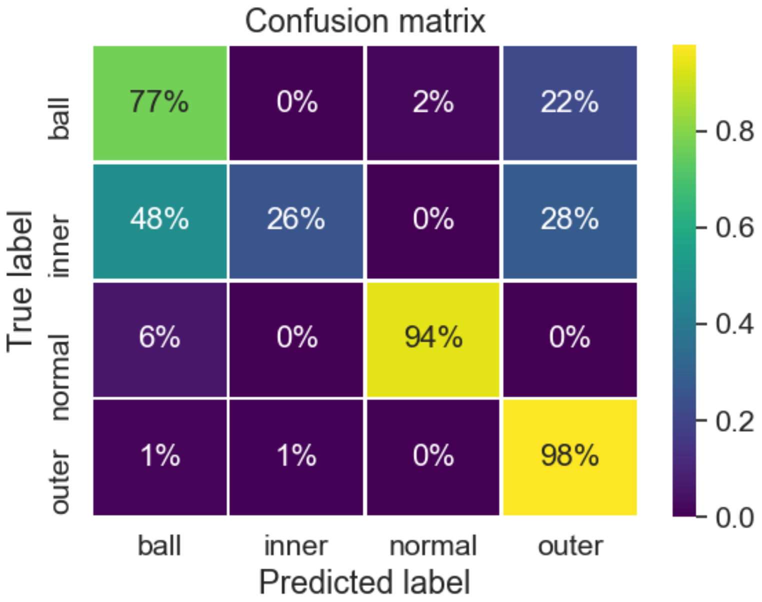

Figure 77: Confusion matrix for the transfer-learning task from fan-end to drive-end. This matrix shows run 2 with an accuracy of 73.43%.

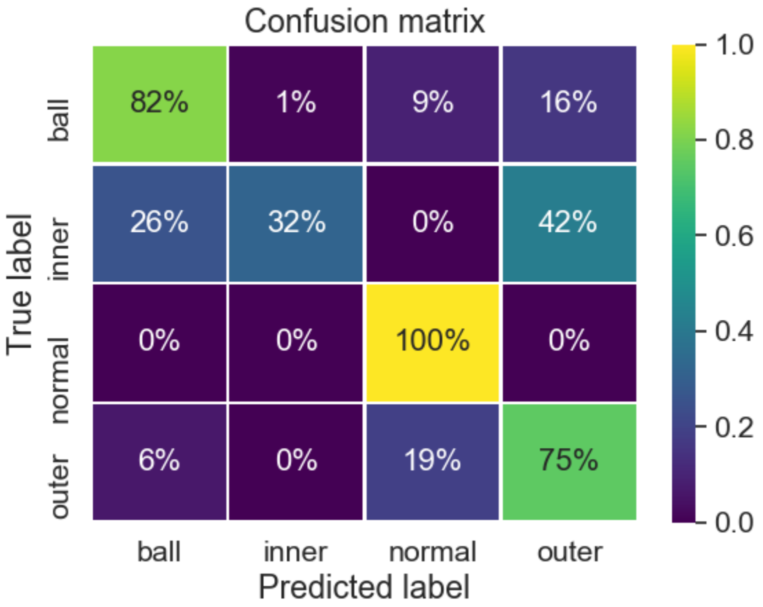

Figure 78: Confusion matrix for the transfer-learning task from fan-end to drive-end. This matrix shows run 3 with an accuracy of 72.31%.

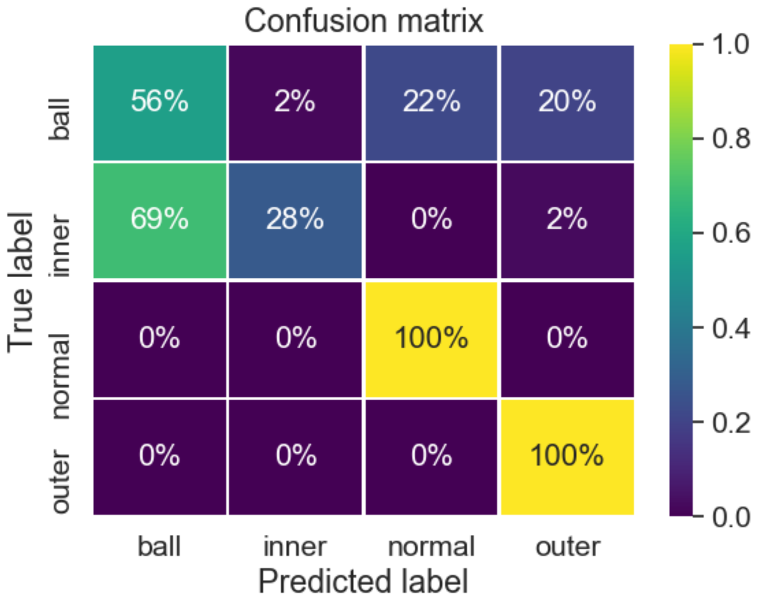

Figure 79: Confusion matrix for the transfer-learning task from fan-end to drive-end. This matrix shows run 4 with an accuracy of 71.09%.

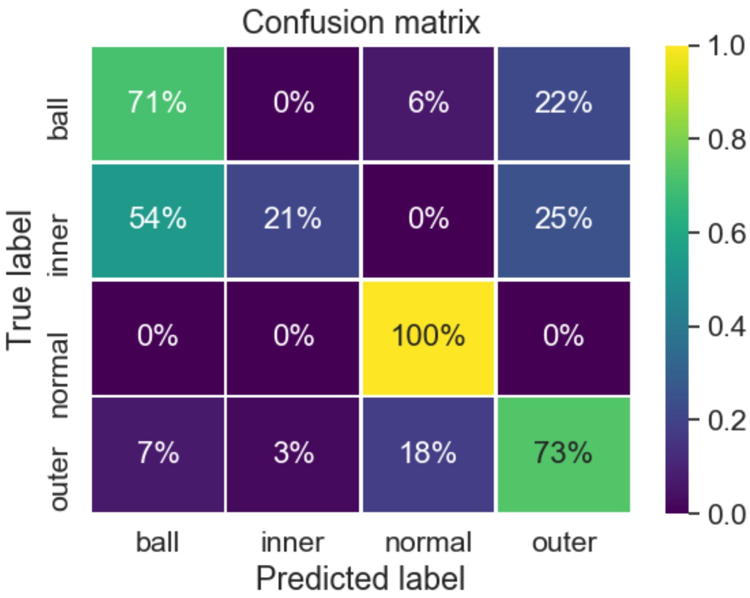

Figure 80: Confusion matrix for the transfer-learning task from fan-end to drive-end. This matrix shows run 5 with an accuracy of 66.15%.

A.5 Verification of the RUL approach

A.5.1 Scoring Algorithm for RUL Estimation Tasks

There is a common metric for validating different RUL algorithms that is based on the percent of prediction error. This metric was also used during the IEEE PHM 2012 Data Challenge [178]. It is based on the relative error (*Er_i*), which is specified by Eq. (27). The parameters of actual RUL (*RUL_Act*) and estimated RUL (*RUL_Est*) are used, while parameter *i* is the index of the test dataset.

$$Er_i = 100 * \frac{RUL\_Act_i - RUL\_Est_i}{RUL\_Act_i} \tag{27}$$

Er is rated in two different ways. Cases in which the calculated RUL is lower than the actual RUL (*Er* > 0) are less severe than cases in which the calculated RUL is longer than the actual RUL (*Er* < 0). In the former case, a component is replaced too early, which only leads to a short, planned downtime and increased material costs, whereas the latter case leads to an unpredicted and thus unplanned failure. For this reason, the weighting is carried out by calculating a score according to Eq.(28).

$$A_i = \begin{cases} e^{-\ln 0.5 * Er_i/5} & \forall\ Er_i \leq 0 \\ e^{-\ln 0.5 * Er_i/20} & \forall\ Er_i > 0 \end{cases} \tag{28}$$

This equation leads to the scoring function shown in Figure 81.

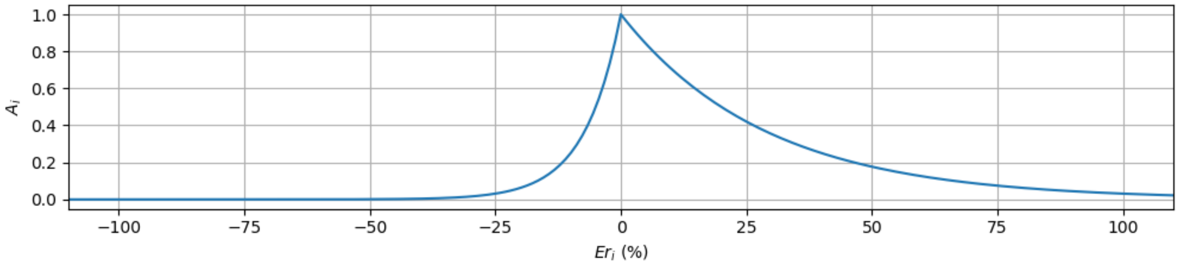

Figure 81: Scoring function A_i as a function of the relative error Er_i. A negative Er_i represents a longer estimated RUL than the actual RUL.

The score for the overall prediction is finally calculated according to Eq. (29), which is the median of the A_is of all *N* test datasets.

$$score = \frac{\sum_{i=1}^{N} A_i}{N} \tag{29}$$

In order to avoid referring to only a "score" in the thesis, this score is also titled as PHM score.

A.5.2 Different Fully Connected Layers

This test scenario is used to benchmark different LSTM and fully connected layer layouts for the RUL task. The test dataset is the FEMTO dataset. The images for the input are created based on 0.2 second time slices and an image size of 64x64 pixels. A time window of 170 measurements was used, but only every second measurement was employed. This leads to the usage of 85 measurements as input. For the training of 300 epochs, a batch size of 120 is used.

The results, which are presented in Table 30, show that layout 2 is the best in terms of PHM score.

Table 30: Different evaluated LSTM layouts. The number of outputs for each layer is given in parentheses. For comparison, the sore of each layout for the RUL task of the IEEE PHM 2012 Data Challenge has been calculated. Layout 2 has the best results.

	Layout 1	Layout 2	Layout 3
Used layers	CNN (8192)	CNN (8192)	CNN (8192)
	LSTM (128)	LSTM (128)	LSTM (128)
	LSTM (64)	LSTM (64)	LSTM (64)
	LSTM (32)	LSTM (32)	Dense (32)
	Dense (1)	Dense (32)	Dropout (rate=0.5)
		Dropout (rate=0.5)	Dense (1)
		Dense (1)	
PHM score	0.1094	0.35	0.05647

A.5.3 Pre-trained Convolutional Layers

This test case is used to state the importance of using network-based transfer learning in the form of transferring pre-trained convolutional layers. For this purpose, the FEMTO dataset was used again. The pre-trained network was trained with samples of the CWRU dataset.

As shown in Figure 82, the PHM score of a network that uses pre-trained convolutional layers is superior.

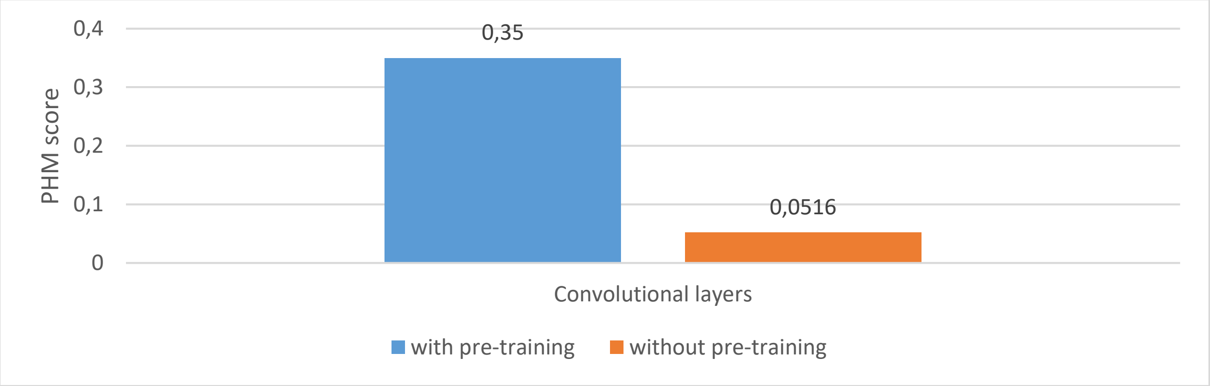

Figure 82: Comparison of the PHM scores between a neuronal network with pertained convolutional layers and one without pre-training. The pre-trained network has a much higher PHM score.