DEPARTMENT OF INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Informatics: Robotics, Cognition, Intelligence

Intelligent Ball Screw Fault Diagnosis using Deep Learning based Domain Adaption and Transfer Learning

Fabian Kolb

DEPARTMENT OF INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Informatics: Robotics, Cognition, Intelligence

Intelligent Ball Screw Fault Diagnosis using Deep Learning based Domain Adaption and Transfer Learning

Intelligente Fehlerdiagnose von Kugelgewinden mittels Deep Learning, basierend auf Domain Adaption und Transfer Learning

Author:	Fabian Kolb
Supervisor:	Zäh Michael; Prof. Dr.-Ing.
Advisor:	Benker Maximilian; M.Sc.
Submission Date:	15.Sept.2022

I confirm that this master's thesis in informatics: robotics, cognition, intelligence is my own work and I have documented all sources and material used.

Munich, 15.Sept.2022 Fabian Kolb

Acknowledgments

Abstract

In order to remain competitive in the ongoing globalization, companies are forced to optimize their productions and processes. Using modern smart technologies like Internet of Things (IoT) and Machine to machine communication (M2M) advances the digitization and automation in production facilities. The immense amount of generated data offers great opportunities for Predictive Maintenance approaches. By analysing machine and production data, machines can be proactively maintained, thereby increasing quality standards and productivity. Since ball screws are one of the most common parts in industrial machines, their health monitoring is especially interesting. Due to strong abrasion ball screws change a lot during lifetime. Therefore, a strong shift in the data distribution can be observed between the working conditions. This makes the fault diagnosis problem more challenging. In order to tackle this problem deep-learning based domain adaption approaches, which are common in the Computer Vision community are promising. In this work, the applicability of such approaches to the given predictive maintenance problem is investigated.

Kurzfassung

Um in der fortschreitenden Globalisierung wettbewerbsfähig zu bleiben, sind Unternehmen gezwungen, ihre Produktionen und Prozesse zu optimieren. Der Einsatz moderner intelligenter Technologien wie Internet of Things (IoT) und Machine-to-Machine-Kommunikation (M2M) treibt die Digitalisierung und Automatisierung in der Produktion weiter voran. Die dabei generierten Datenmengen ermöglichen moderne für Predictive Maintenance-Ansätze. Durch die Analyse von Maschinen- und Produktionsdaten können Maschinen proaktiv gewartet werden, um dadurch die Qualitätsstandards und Produktivität zu erhöhen. Da Kugelgewindetriebe in Industriemaschinen häufig verbaut werden, ist deren Zustandsüberwachung besonders interessant. Durch den hohen Verschleiß verändern sich Kugelgewindetriebe über ihrer Lebensdauer stark. Zwischen den Betriebszuständen ist deshalb eine starke Verschiebung in der Datenverteilung zu beobachten. Dies macht die Fehlerdiagnose herausfordernd. Deep-Learning-basierte Domain-Adaption Ansätze, die in der Computer-Vision-Community verbreitet sind, sind daher vielversprechend. In dieser Arbeit wird die Anwendbarkeit solcher Ansätze für das gegebene Predictive Maintenance Probelem untersucht.

Contents

1. Introduction

Internet of Things (IoT) and data-driven techniques have revolutionized the industry. Cloud computing applications collect huge amounts of manufacturing data. As a key role in modern production facilities, predictive maintenance applications have fully embraced the Big Data revolution. By analyzing production and machine data future machine failures or working conditions can be predicted. This is especially helpful to reduce downtime of single machines or whole production lines [1].

In a wide range of industrial machines ball screws are used, which translate rotatory into linear motion with high precision. During longtime use degradation lowers the stiffness within the ball screw components. This leads to an inaccurate position movement, which lowers the precision of the machine and therefore the production quality. In the worst case a whole machine failure may occur. For this reason, tracking the ball screw health condition is of great importance. In recent years, Predictive Maintenance research mainly focused on rolling bearings and gearboxes. Since industrial machines and components have different stages of operation, working cycles and installed sensors, requirements for the predictive maintenance algorithms vary. The transferability of predictive maintenance algorithms is difficult and does usually require adjustments. However, intelligent data-driven methods have been successfully developed. Unfortunately, a lot of these approaches assume monitoring data used for training and testing to be from the same data distribution. In reality this is not the case. Since the operating conditions, component degradation and machine settings may change, the fault characteristics of the machines do also change. Therefore, the fault diagnosis system which is trained on a limited amount of data can not generalize well on the testing data. Consequently, the diagnosis system might have an unsatisfactory performance in the real industrial scenario. This phenomena is called domain shift problem [2].

As shown in fig. 1.1 health monitoring traditionally is restricted to physical-based or conventional data-driven approaches. The persistence of physical laws to satisfactorily explain the underlying complexity of a machine sounds like a promising way for predictive maintenance. Besides that these models do not require any historical fault data [3]. Unfortunately in this scenario the physics we know needs to rely on data which is often perturbed. Not including these perturbations in the models reduces the performance. With the establishment of modern sensors, networks and computing systems data-driven approaches became into the focus of research. In conventional data-driven approaches traditional hand-crafted features are used to extract expressive information from the data in a first step. Subsequently a shallow classifier is responsible for an accurate prediction. Both just mentioned methods suffer from two problems. Firstly, in complex real-world problems establishing physical-based models or conventional hand-crafted features is quiet a laborious task and expect a lot of experience and human labor. Secondly, an in-line update with measured data is not possible. The

physical-based models and hand-crafted features are restricted to the application of interest and its transferability is very limited. Therefore, the limited flexibility and effectiveness of the two traditional approaches is responsible for the growing interest in deep learning based Predictive Maintenance. Deep neural networks can capture relations within complex and high-dimensional data. By using multiple layers, neural networks can progressively extract higher-level features from the raw input. Due to the automatic learning of neural networks they can be adjusted quickly to different problems. Especially due to the increased amount of data, computation power and huge research the popularity of deep learning increased a lot. [1] [2].

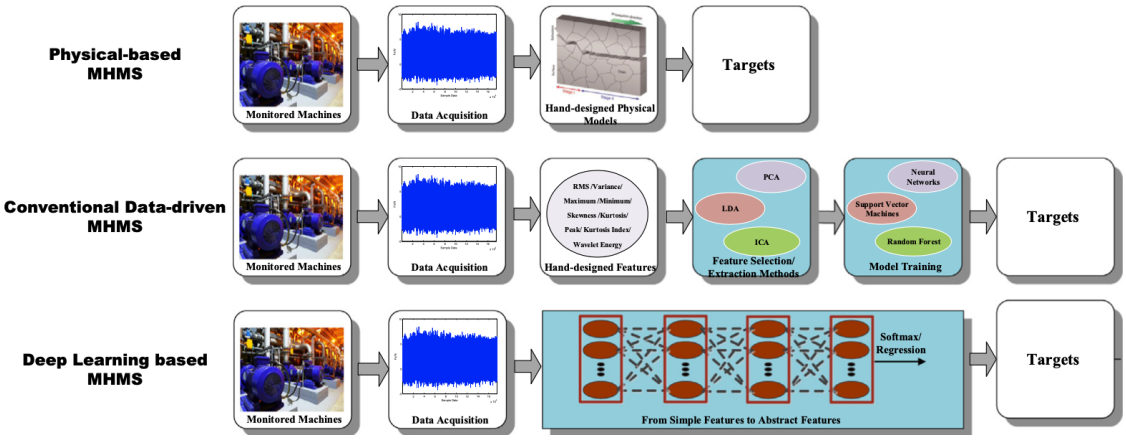

Figure 1.1.: Physical Models, Conventional Data-driven Models and Deep Learning Models for predictive maintenance [1]

In the computer vision community domain adaption and transfer-learning techniques recently received more and more attention. Knowledge learned from the supervised training data, denoted as source domain, is transferred to the unsupervised testing data, denoted as target domain. The target and source domain data come from different problems which must be related and structured similarly. Generally transfer learning approaches can be grouped in four different types.

- **Instance-based transfer learning** where different weights are assigned to data samples from the source domain. These weights are proportional to the similarity between sample and target domain.
- **Feature-based transfer learning** which has the goal to find a feature space in which the discrepancy between the domains is reduced.
- **Model-based transfer learning** which uses classifiers that can be transfered directly or fine-tuned to make the model perform well on the target domain.
- **Relation-based transfer learning** where the goal is to find and utilize similarities between the two domains to transfer knowledge [2].

In order to compensate the domain shift problem in health monitoring tasks, domain adaption and transfer learning approaches are promising. Since the data from the source and target domain include the same machine health conditions and shared underlying features exist between them, domain adaptation approaches to compensate the domain shift are investigated. Fig.1.2 illustrates how feature-based domain adaption can be used to find a cross-domain classifier which accurately separates source and target domain data in a domain invariant feature space [4].

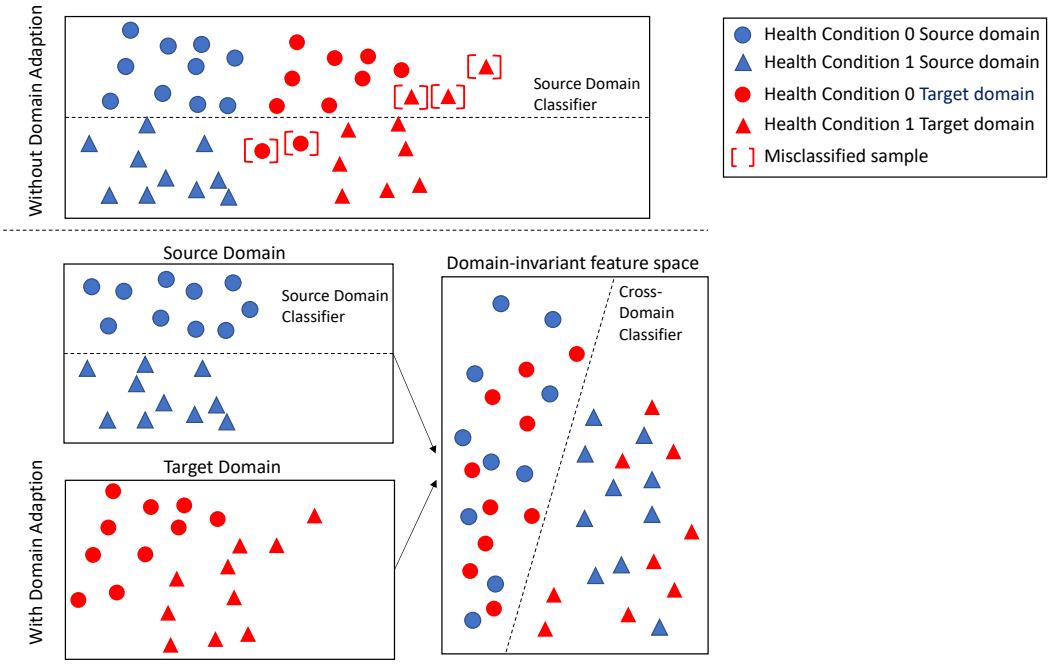

Figure 1.2.: Domain adaption for health monitoring of machines with different working conditions [4]

1.1. Section

Citation test (with Biber) [5].

1.1.1. Subsection

See Table 1.1, Figure 1.3, Figure 1.4, Figure 1.5, Figure 1.6, Figure 1.7.

This is how the glossary will be used.

Donor dye, ex. Alexa 488 (D_{dye}), Förster distance, Förster distance (R_0), and k_{DEAC}. Also, the TUM has many computers, not only one Computer. Subsequent acronym usage will only print the short version of Technical University of Munich (TUM) (take care of plural, if

Table 1.1.: An example for a simple table.

A	B	C	D
1	2	1	2
2	3	2	3

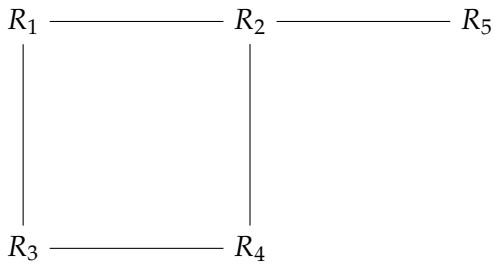

Figure 1.3.: An example for a simple drawing.

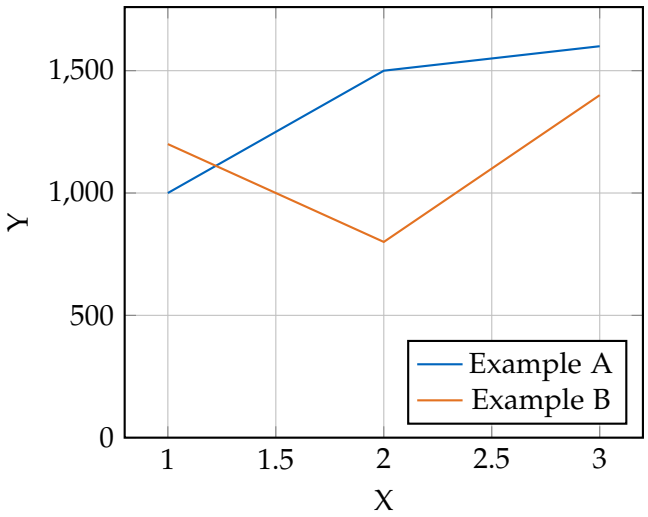

Figure 1.4.: An example for a simple plot.

```
SELECT * FROM tbl WHERE tbl.str = "str"
```

Figure 1.5.: An example for a source code listing.

Figure 1.6.: Includegraphics searches for the filename without extension first in logos, then in figures.

Figure 1.7.: For pictures with the same name, the direct folder needs to be chosen.

(a) The logo.

(b) The famous slide.

Figure 1.8.: Two TUM pictures side by side.

needed!), like here with TUM, too. It can also be –> hidden[1] <–.

[(TODO: Now it is your turn to write your thesis.
This will be a few tough weeks.)]
[(Done: Nevertheless, celebrate it when it is done!)]

[1]Example for a hidden TUM glossary entry.

2. Related Works

Use with pdfLaTeX and Biber.

2.1. Section

Citation test (with Biber) [5].

2.1.1. Subsection

3. Methods

3.1. Maximum Mean Discrepancy

Maximum Mean Discrepancy (MMD) is a criterion which estimates the discrepancy between two distribution. MMD can be used to optimize the network such that the distribution discrepancy is reduced in a data domain-invariant feature space. The discrepancy is measured as squared distance between the distribution kernel embeddings in the reproducing kernel Hilbert space (RKHS). The distribution discrepancy across domains is measured in the layers of the neural network in order to avoid feature transferability degradation. One has to pay attention to not transfer noise or irrelevant information. This destroys the structure of the source and target domain data and therefore makes the classification task even more difficult [6].

$$M_k(P,Q) = \left| \boldsymbol{E_P}[\Phi(\boldsymbol{X^s})] - \boldsymbol{E_Q}[\Phi(\boldsymbol{X^t})] \right|^2_{Hk} \tag{3.1}$$

Hk denotes the RKHS, which is described by the characteristic kernel k and the mapping function Φ. Taking the identity function as mapping function results in matching the distribution means. When using more complex mapping functions also higher order moments can be matched [7]. The distributions of the source domain $X^s = \{x_i^s\}_{i=0,\dots,n_s}$ and target domain $X^t = \{x_i^t\}_{i=0,\dots,n_t}$ are represented by P and Q. $\boldsymbol{E_p}[.]$ is the expected value of the source distribution P in the feature space. The kernel choice is of great importance when applying MMD. For this reason it makes sense to combine several kernels in order to profit from their individual performance [6].

$$k(\boldsymbol{X^s}, \boldsymbol{X^t}) = \sum_{i=0}^{N_k} k_{\sigma_i}(\boldsymbol{X^s}, \boldsymbol{X^t}) \tag{3.2}$$

N_k denotes the number of kernels used in the the RKHS and k_{σ_i} represents one individual RBF kernels. Also, other kernels like linear kernels could be used, but current research shows that RBF kernels usually perform best [2]. In our attempt we used 5 RBF kernels with the bandwidth parameters 1, 2, 4, 8, 16.

3.1.1. Proposed training with MMD and cross entropy loss

In the following the training of deep-learning based domain adaption models which use a source cross-entropy and MMD loss is explained and visualized in fig. 3.2. In the given

domain adaption task the model processes data from the labeled source and unlabeled target domain. In a first step the data is preprocessed and split into several smaller sequences. Each sequence is fed into the model. A 1D CNN extracts expressive features and subsequently a fully connected layers predicts the health condition classes of each sample. The MMD loss estimates the discrepancy between latent feature vectors of source and target domain samples in several layers of the neural network. Source and target domain samples are groped up randomly in pairs of two. For each pair the MMD loss is calculated in several hidden layers and added up. By applying a source cross-entropy loss the model learns to predict correct labels for the source domain samples. Fig. 3.1 symbolically shows how the MMD as well as the source cross-entropy loss is extracted from the model.

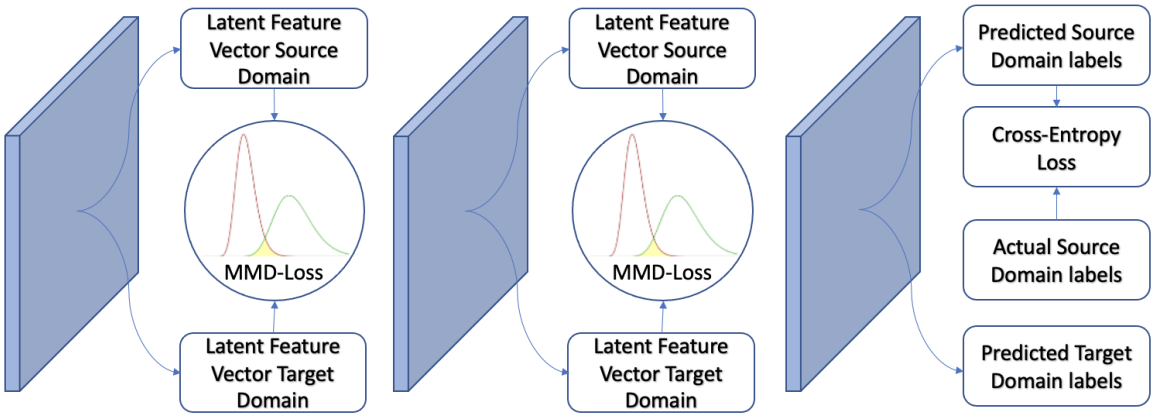

Figure 3.1.: Cross Entropy Loss and MMD Loss in Neural Networks

The cross entropy loss is used to solve the classification problem and the MMD loss to reduce the domain discrepancy. When including several losses, a weighted average must be defined in order to train neural networks. Often these weighting factors are hyperparameters which need to be defined before training. The following equation presents the weighted cross entropy and MMD loss average which is used to optimize the model:

$$\text{Total Loss} = \text{Source Cross-Entropy Loss} + \text{GAMMA} * \text{MMD Loss} \tag{3.3}$$

In general the training is repeated until the maximum number of epochs is reached. After the training is completed the model can be used to predict the health labels for the target domain data.

3.1.2. Dummy Dataset

In order to optimize the model to extract more domain invariant features a MMD loss can be applied. In this work simple dummy example was established to evaluate the effect of

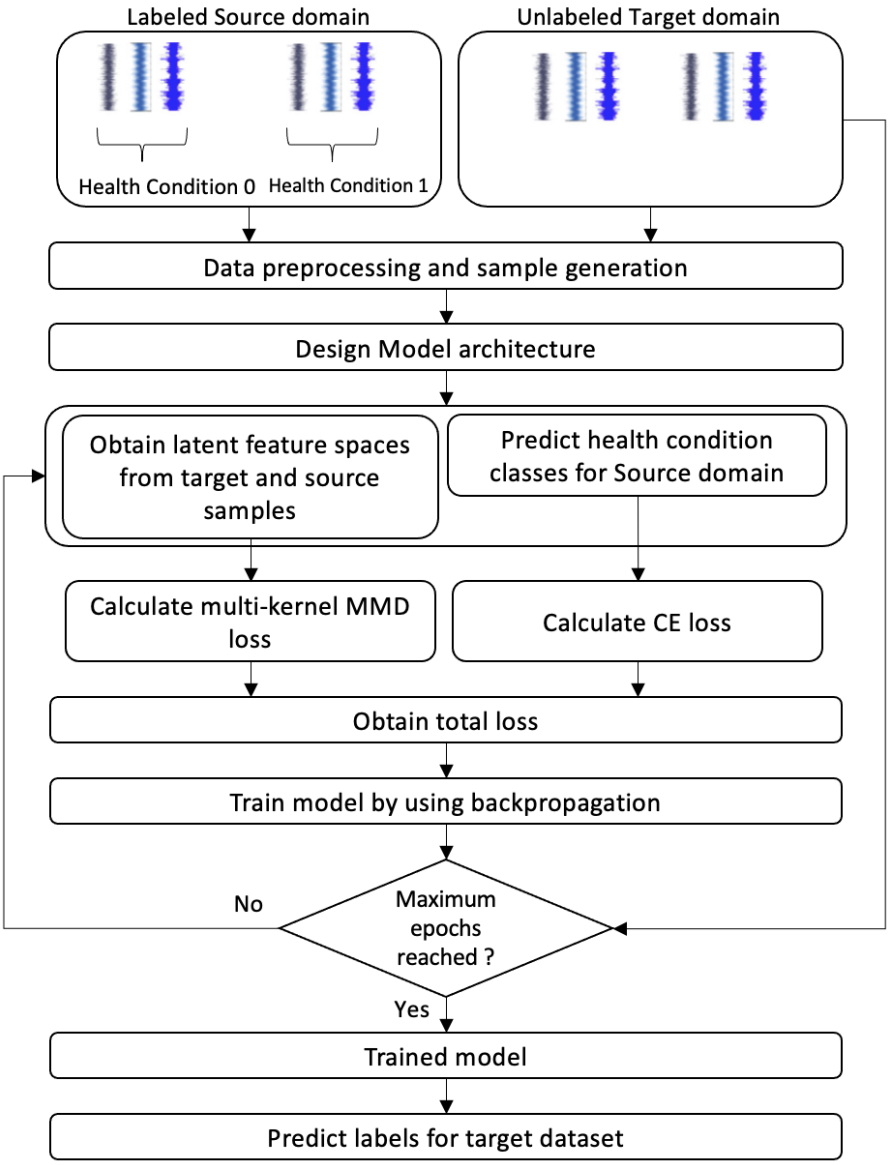

Figure 3.2.: Traing of model

the MMD loss on the feature extraction. Since one has to deal with irregularities, outliers and noise in real-world data it is helpful to study the MMD loss on a self-made dataset which is not disturbed by these effects. Similarly to predictive maintenance applications the model in the dummy example processes time sequences. In the dummy example the time sequences consist of 1000 data points which are sampled from a cosinus curve. In order to create a dataset which simulates a classification problem with two classes and two domains the dataset samples from four cosinus curves with characteristic amplitude and frequency. By adjusting amplitude and frequency the domain adaption problem can be made more or less difficult. In order to sample several differing sequences for a class and domain the sampling process must include a certain randomness. For each sampling iteration the domain- and class-specific amplitude and frequency of the cosinus curve is perturbed within a given boundary. This changes the underlying characteristic of each sample. Besides that noise is added on each of the 1000 datapoint within the sequence. This is necessary to to generate a periodic signal which is closer to noisy real-world vibration signals. For the sake of simplicity the dataset contains just one dimensional sequencies which simulate one feature to process. In Fig. 3.3 four data sequences are shown representing one example sequence for each class and domain.

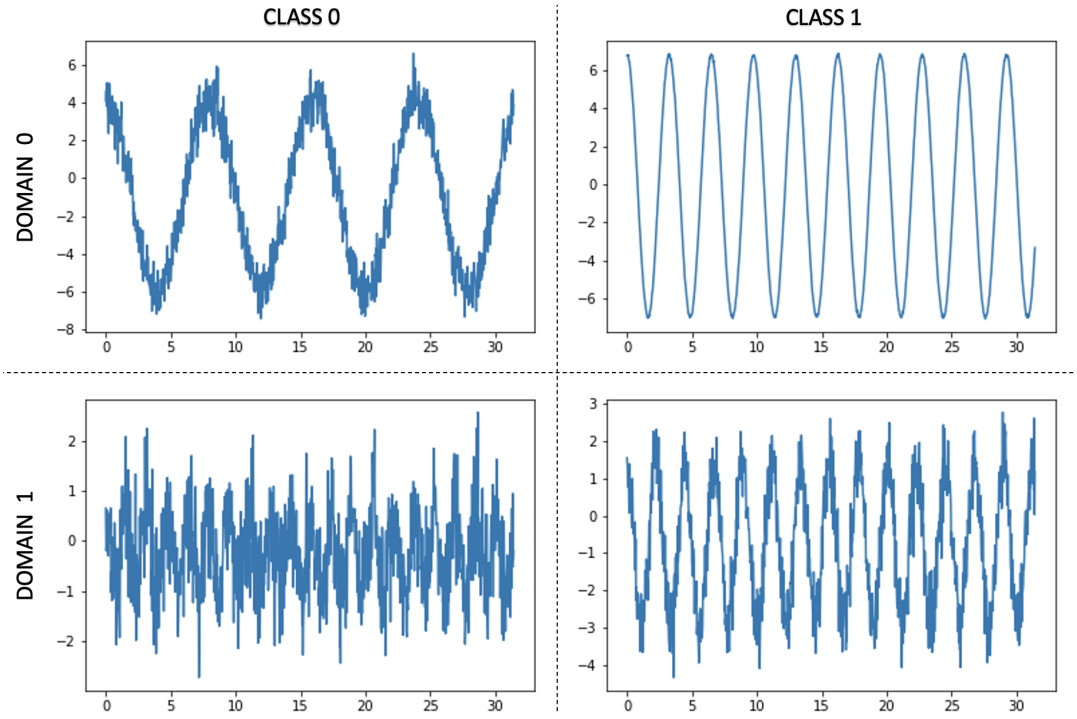

Figure 3.3.: Data Window Samples for each domain and class

In fig. 3.4 one can see how the applied perturbation and noise during the sampling process changes the data sequences belonging to the same class and domain.

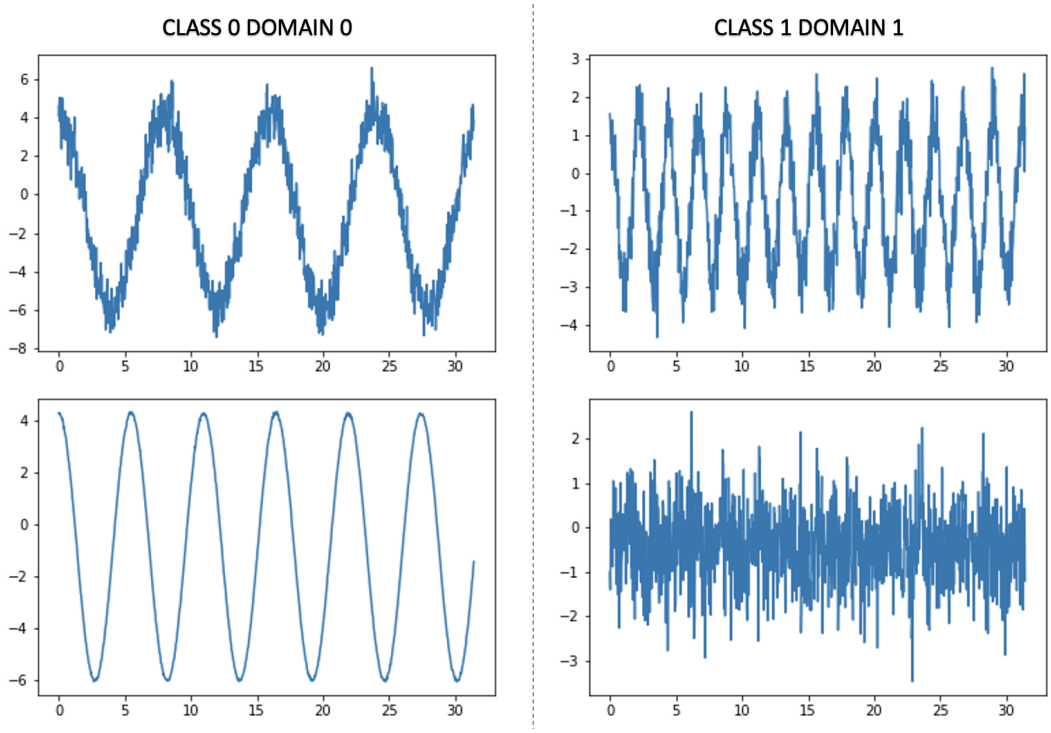

Figure 3.4.: Influence of perturbation when sampling several data sequences for one class and domain

3.1.3. Ball Screw Dataset

In order to evaluate different predictive maintenance approaches, data from a DMG DMC 55H duoblock milling machine of the meanwhile German-Japanese manufacturer DMG Mori were collected. Particularly, the predicting the health condition of the linear guiding shoes (LGSs) and the ball screw drives (BSDs) are of interest. The different LGS classes are specified by the preload values and the occurrence of pitting damages. The health condition classes are specified by one letter and two digits. An existing pitting damage is indicated by the letter "P" and otherwise by the letter "C". The first digit represents the preload class and the second digit the observation within that class. During the experiment for LGSs are mounted in parallel all the time. Just three health condition classes exist since pitting does not exist in these components. Via TNC Scope internal control data and via TNC Opt internal control data is accessed. Three triaxial Kistler 8762A10 piezo-eletric accelerometers were used to track accelerations in the three spatial directions. A defined test cycle was used to track the data in order to keep the reproducibility of this experiment, which is described in fig. 3.5. Machine data is collected during constant speed, direction change and sweep excitement along the machine tools X-, Y-, Z-axis separately. During the constant speed excitement the machine tools are moved along the whole axis (Δx = 600mm) back and forth. In the direction change excitation the movement of the machine tools is restricted to a small part of the axis (Δx = 1mm) and the directions are changed with a high frequency. In the sweep excitement the motor, moving the machine tools along the different axis, receives a target speed in form of a sine sweep. Before recording data, the machine is warmed up in order to create equivalent circumstances for each run. For this reason constant speed excitement is applied for 60 min.

In total 49 features are recorded during one sub cycle and stored as individual sequences. The recording of these features is described by table. Table. 3.1

name	sensor	frequency	samples
C:s ist/X	TNC Scope	10 kHz	75000
C:s soll/X	TNC Scope	10 kHz	75000
C:s diff/X	TNC Scope	10 kHz	75000
C:v (n ist)/X	TNC Scope	10 kHz	75000
C:v (n soll)/X	TNC Scope	10 kHz	75000
C:P mech./X	TNC Scope	10 kHz	75000
C:Pos. Diff./X	TNC Scope	10 kHz	75000
C:I ist/X	TNC Scope	10 kHz	75000
C:I soll/X	TNC Scope	10 kHz	75000
C:x bottom	Acc	10 kHz	75000
C:y bottom	Acc	10 kHz	75000
C:z bottom	Acc	10 kHz	75000
C:x nut	Acc	10 kHz	75000
C:y nut	Acc	10 kHz	75000
C:z nut	Acc	10 kHz	75000

C:x top	Acc	10 kHz	75000
C:y top	Acc	10 kHz	75000
C:z top	Acc	10 kHz	75000
D:s ist/X	TNC Scope	10 kHz	75000
D:s soll/X	TNC Scope	10 kHz	75000
D:s diff/X	TNC Scope	10 kHz	75000
D:v (n ist)/X	TNC Scope	10 kHz	75000
D:v (n soll)/X	TNC Scope	10 kHz	75000
D:P mech./X	TNC Scope	10 kHz	75000
D:Pos. Diff./X	TNC Scope	10 kHz	75000
D:I ist/X	TNC Scope	10 kHz	75000
D:I soll/X	TNC Scope	10 kHz	75000
D:x bottom	Acc	10 kHz	75000
D:y bottom	Acc	10 kHz	75000
D:z bottom	Acc	10 kHz	75000
D:x nut	Acc	10 kHz	75000
D:y nut	Acc	10 kHz	75000
D:z nut	Acc	10 kHz	75000
D:x top	Acc	10 kHz	75000
D:y top	Acc	10 kHz	75000
D:z top	Acc	10 kHz	75000
S:x bottom	Acc	10 kHz	153601
S:y bottom	Acc	10 kHz	153601
S:z bottom	Acc	10 kHz	153601
S:x nut	Acc	10 kHz	153601
S:y nut	Acc	10 kHz	153601
S:z nut	Acc	10 kHz	153601
S:x top	Acc	10 kHz	153601
S:y top	Acc	10 kHz	153601
S:z top	Acc	10 kHz	153601
S:Nominal rotational speed	TNC opt	1 kHz	16384
S:Actual rotational speed	TNC opt	1 kHz	16384
S:Actual position of the position encoder(dy/dt)	TNC opt	1 kHz	16384
S:Actual position of the motor encoder(dy/dt)	TNC opt	1 kHz	16384

Table 3.1.: feature description of the 49 different time-series

Due to abrasion the different parts wear down. LGSs are separated in three and the BSDs in four different health condition classes. Usually the lifteime of ball screws is shorter than

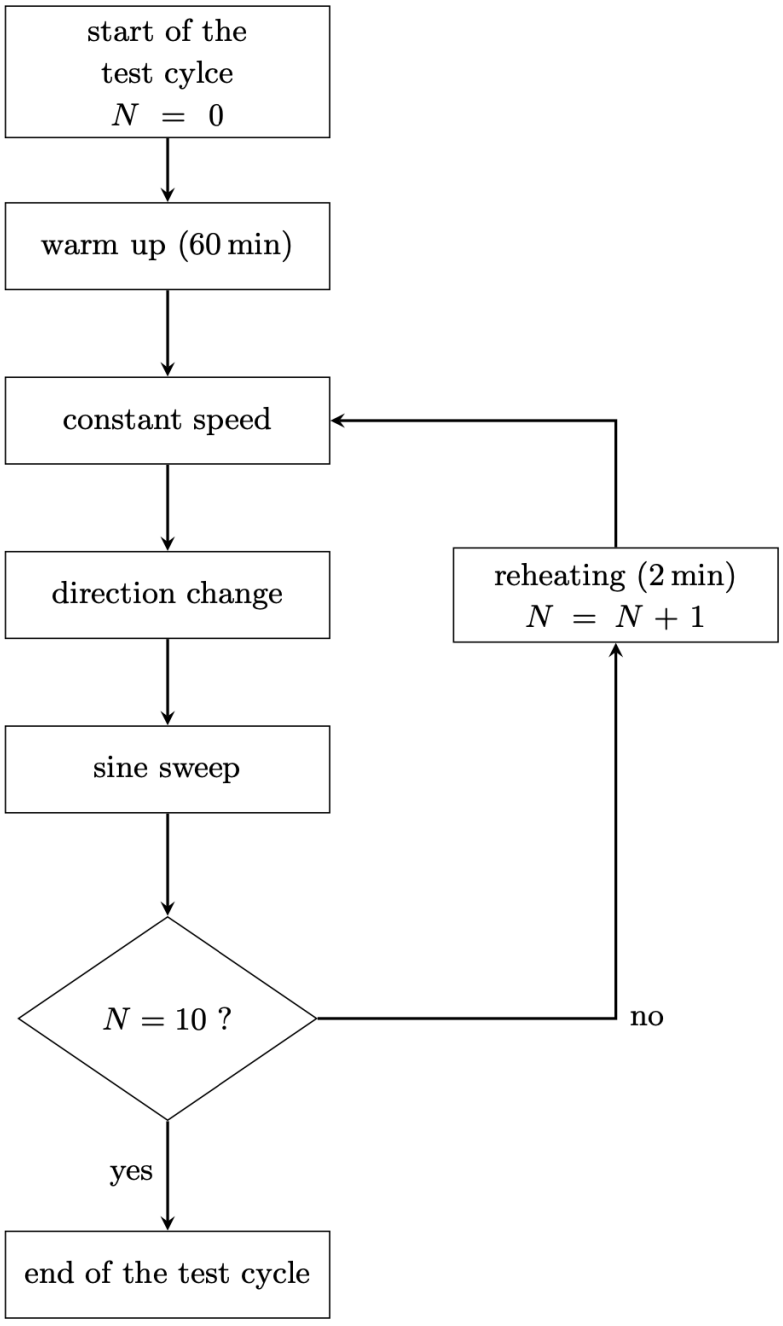

Figure 3.5.: Flow chart explaining the test cycle used in the presented dataset

that of the profile rails. This means that ball screws need to be replaced in shorter internals. Different combinations of health condition classes from profile rails and ball screws were recorded, which are shown in the following table. 3.2

		BSD								
		C31	**C21**	**C11**	**P1**	**C22**	**C12**	**C32**	**C33**	**P2**
LGS	**C3**	1	2	3	4	5	6	7	8	9
	C2	10	11	12	13	14	15	16	17	18
	C1	18	19	20	21	23	24	25	26	27

Table 3.2.: Recorded combinations of LGS and BSD health conditions

In order to test the domain adaption approaches the dataset is split in source and target domains. The source domain consists of the health condition states 1,2,3,4,10,11,12,13,18,19,20,21 and the target domain of 5,6,7,9,14,15,16,18,23,24,25,27. The models used in the experiments should be trained to predict the BSD health condition states accurately. To reduce the complexity of the problem the number of BSD health conditions was reduced to BSD health condition classes C1 and C2 (1,2,5,6,10,11,14,15,18,19,23,24). The dataloader used to prepare the dataset takes the data and separates them in shorter sequences of length 1024. These sequences are fed to the model as a single sample. When using several of the 49 recorded features the sequences are cleaned from Nan values and synchronized. The dataset is randomly separated in a train and validation part with a split of 80%/20%.

Model

In the MMD experiments different model architectures and optimizer were used. Generally the model consist of a 1D CNN which is responsible to extract expressive features. The CNN consists of 3 convolutional layers. With increasing depth of the network the kernel size decreases. This helps to extract more general and global features in shallow and more specific and local features in deeper layers. Depending of the underlying data complexity in the different experiments max pooling layers are included in order to avoid overfitting and exploding gradients. Besides that the generalization well to the target domain. Including batch normalization after convolutional layers helps to make the training faster and more stable. This is done by fixing the means and variances of the layers inputs for each batch. After iteratively applying these three types of layers the output of the CNN is flattend to a 1D vector. This vector is used as an input for the subsequent classifier. The latent space size of the Classifier is reduced in each layer. The 2D output of the neural network represents the probability for the labels of the two health condition classes.

Balancing Cross-Entropy and MMD loss

In the following section the influence of weighting factor GAMMA on the training success is evaluated. The CNN of the model presented in fig. 3.6 which acts as feature extractor is optimized.

```
CNN(
  (conv1): Conv1d(1, 64, kernel_size=(100,), stride=(1,))
  (conv2): Conv1d(64, 32, kernel_size=(10,), stride=(1,), padding=(1,))
  (batch1): BatchNorm1d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (conv3): Conv1d(32, 32, kernel_size=(5,), stride=(1,), padding=(1,))
  (batch2): BatchNorm1d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
)
Classifier(
  (fc1): Linear(in_features=28544, out_features=100, bias=True)
  (fc2): Linear(in_features=100, out_features=3, bias=True)
  (fc3): Linear(in_features=3, out_features=2, bias=True)
)
```

Figure 3.6.: Data distribution: Influence of GAMMA to the model training with MMD loss

The data distribution of the source and target domain samples in the 3D penultimate hidden layer of the classifier is visualized in fig. 3.8. Fig. 3.7 shows the development of the MMD and cross entropy loss throughout the training process. When picking a small GAMMA, the model suffers from a bad separability of the two classes across the two domains. In this case the source cross-entropy loss dominates the training. Instead of reducing the domain discrepancy the model training focuses solely on predicting source samples correctly. When picking a big GAMMA the training is dominated by the MMD loss such that the correct prediction of source domain samples becomes irrelevant. Since the target labels are unknown, the MMD loss is calculated between source and target samples of the same and different classes. The MMD loss reduces the inter and intra class distance between the latent space feature vectors of source and target samples. The separability of the classes is reduced. A trivial solution for this optimization problem is a model which processes all samples such that they collapse at the same point in the latent feature space.

Just when the GAMMA is chosen correctly the source cross-entropy and MMD loss can be reduced. Is the GAMMA too big or too small the optimization just focuses on one of the two losses. In the case of a big GAMMA, the model does not learn to separate between classes, but to reduce the distance between the samples of all classes and domains. If GAMMA is picked to small the model is able to predict the the labels for the source domain samples accurately but can not transfer that knowledge on the target domain.

Differences of labeled and unlabeled MMD loss

The idea behind domain adaption is to aggregate knowledge while solving one problem and transferring that knowledge to another problem. For this reason in domain adaption tasks the goal is to restrict the supervised learning solely on the source domain data. Section 3.1.3

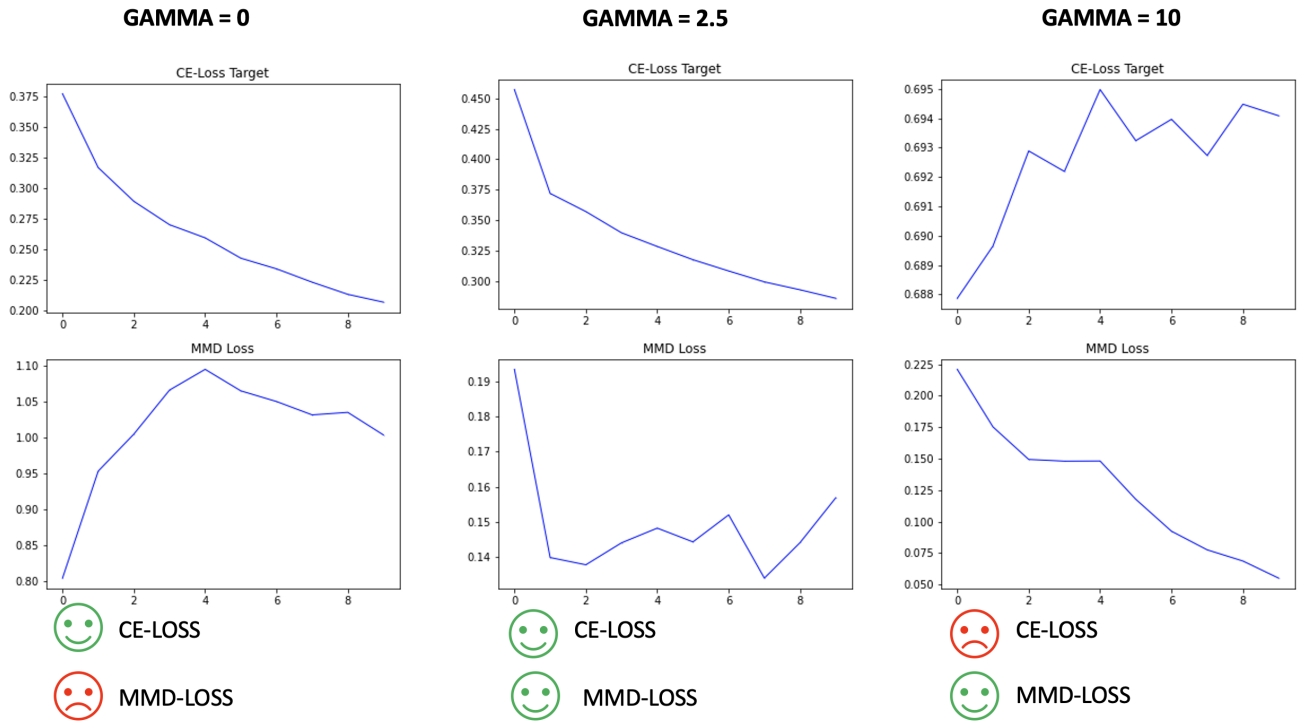

Figure 3.7.: Learning curves: Influence of GAMMA to the model training with MMD loss

describes that applying the MMD loss in general helps to reduce the domain discrepancy. Since the target labels are unknown the intra and inter class distance between source and target samples is minimized. Obviously this also reduces the separability of classes in the source and target domain. In the literature domain adaption approaches which use a small amount of the target domain labels are known under the name "Few-shot transfer learning" [8]. Similarly in this section the effect of including a few target labels into the training is analysed. The cross-entropy loss is still restricted to source samples and corresponding labels only. Solely the MMD loss is allowed to use the target labels. This allows to separately calculate the distance between source and target samples of similar and of different classes. The labeled MMD loss optimizes the model such, that the intra class distance between the domains is maximized and the inter class distance is minimized. The domain discrepancy is reduced by minimizing the the inter class distance. The separability is improved by maximizing the the intra class distance. In parallel the training still includes the source cross-entropy loss in order to improve the source domain classification task. The hyperparameter GAMMA intra class and GAMMA inter class are used to balance the training scope of reducing the inter class distance, maximizing the intra class distance and improving the source domain classification problem:

$$\begin{aligned}\text{Total MMD Loss} = \text{GAMMA intra class} * \text{MMD loss intra class} \\ + \text{GAMMA inter class} * \text{MMD loss inter class} + \text{CE Loss}\end{aligned} \tag{3.4}$$

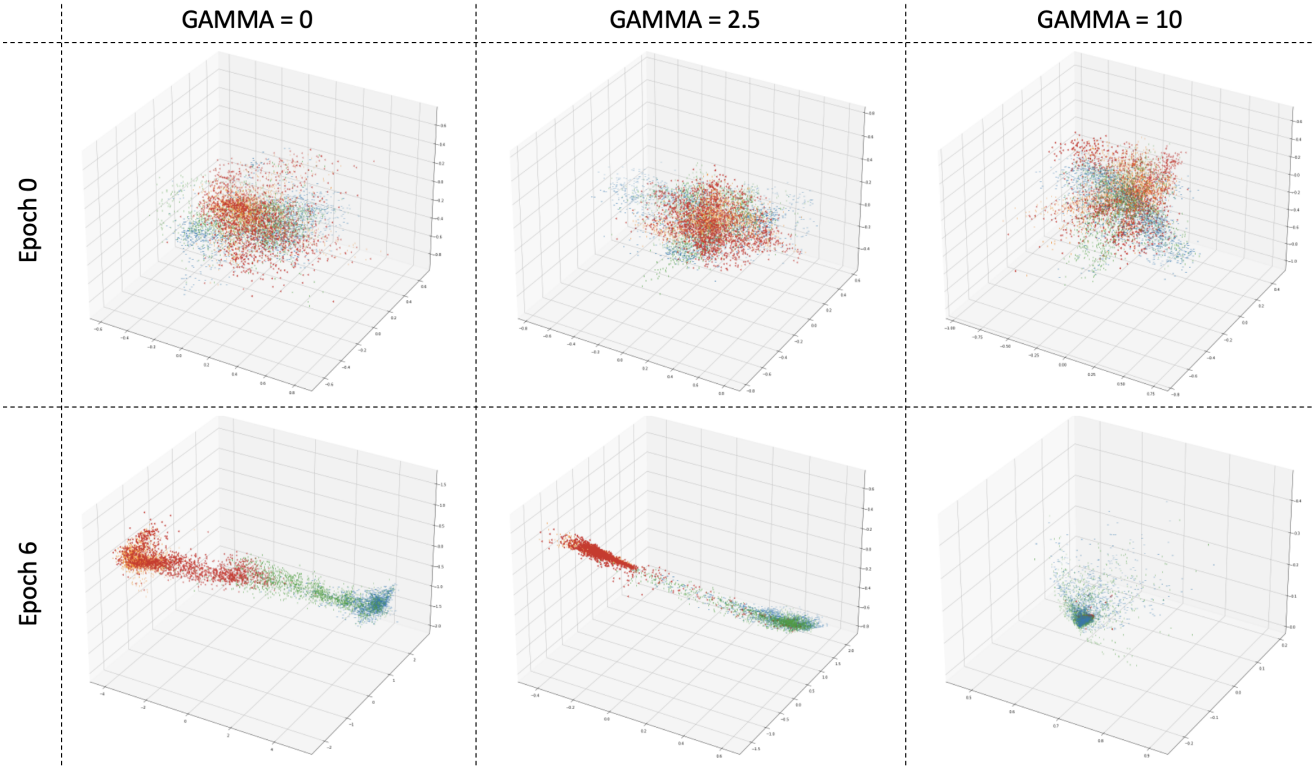

Figure 3.8.: Data distribution: Influence of GAMMA to the model training with MMD loss

The MMD loss which includes the target labels is called "labeled MMD loss" and otherwise "unlabeled MMD loss". For the evaluation the CNN of the model presented fig. 3.6 is optimized again. Fig. 3.9 visualizes the distribution of the latent feature representation distribution of the source and target samples within the penultimate hidden layer. Throughout the training both MMD losses reduce the domain discrepancy and increase the separability between classes of both domains. When applying the labeled MMD loss the distance between classes of both domains is bigger compared to the unlabeled MMD loss. This makes classification problem easier and besides that the trivial solution in which the latent feature representation of all samples collapse at one point is prevented. On the other hand one has to know that target labels of just 20 percent are known. Therefore, the labeled MMD loss can just be applied on a small subset of coupled source and target samples. In comparison the unlabeled MMD loss can be applied on the whole dataset.

Latent feature space choice for MMD loss

This section analyses how calculating the source and target domain discrepancy with the features of different hidden layers influences the model training. Fig. 3.10 visualizes the calculation of two MMD losses extracting different latent features from the classifier and the CNN. By training the model presented in fig. 3.11 the two MMD losses are compared. The "regular FC MMD" loss calculates the source and target discrepancy from the latent features in the fully connected layers of the classifier. The "regular FC CNN MMD" loss additionally considers the feature maps extracted right after the convolutional layers in the CNN.

The development of the accuracy, the source cross-entropy and MMD loss during the training process are shown in fig. 3.12 and fig. 3.13. The accuracies in source and target domain tend to be a little higher when including the CNN feature maps into the MMD loss. The main advantage of including the CNN feature maps is the increased stability of the training. The MMD and cross entropy loss as well as the accuracies converge faster and smoother.

By passing data through the model, features with varying levels of abstraction are extracted. From the just presented results it seems reasonable to supervise the domain discrepancy in feature spaces of different abstraction levels. Generally deeper layers in neural network extract rather more task specific features. Therefore deeper layers often suffer more from domain-dependencies. Since each hidden layers output influences subsequent layers, it makes sense to apply the MMD loss early in the network. Reducing the domain discrepancy in shallow layer makes the network extract more domain-invariant features in deeper layers. Without intervening in early layers the domain discrepancy gets more difficult to solve in deeper layers. Reducing the domain discrepancy in the final layers of the model leads to an less stable optimization problem. In this case the MMD and source cross-entropy loss show tendencies to work against each other. In these cases the optimization seems to focus just on one of optimization goals . Calculating the regular FC CNN MMD loss is quiet expansive since the features extracted from the convolutional layers are complex and high-dimensional.

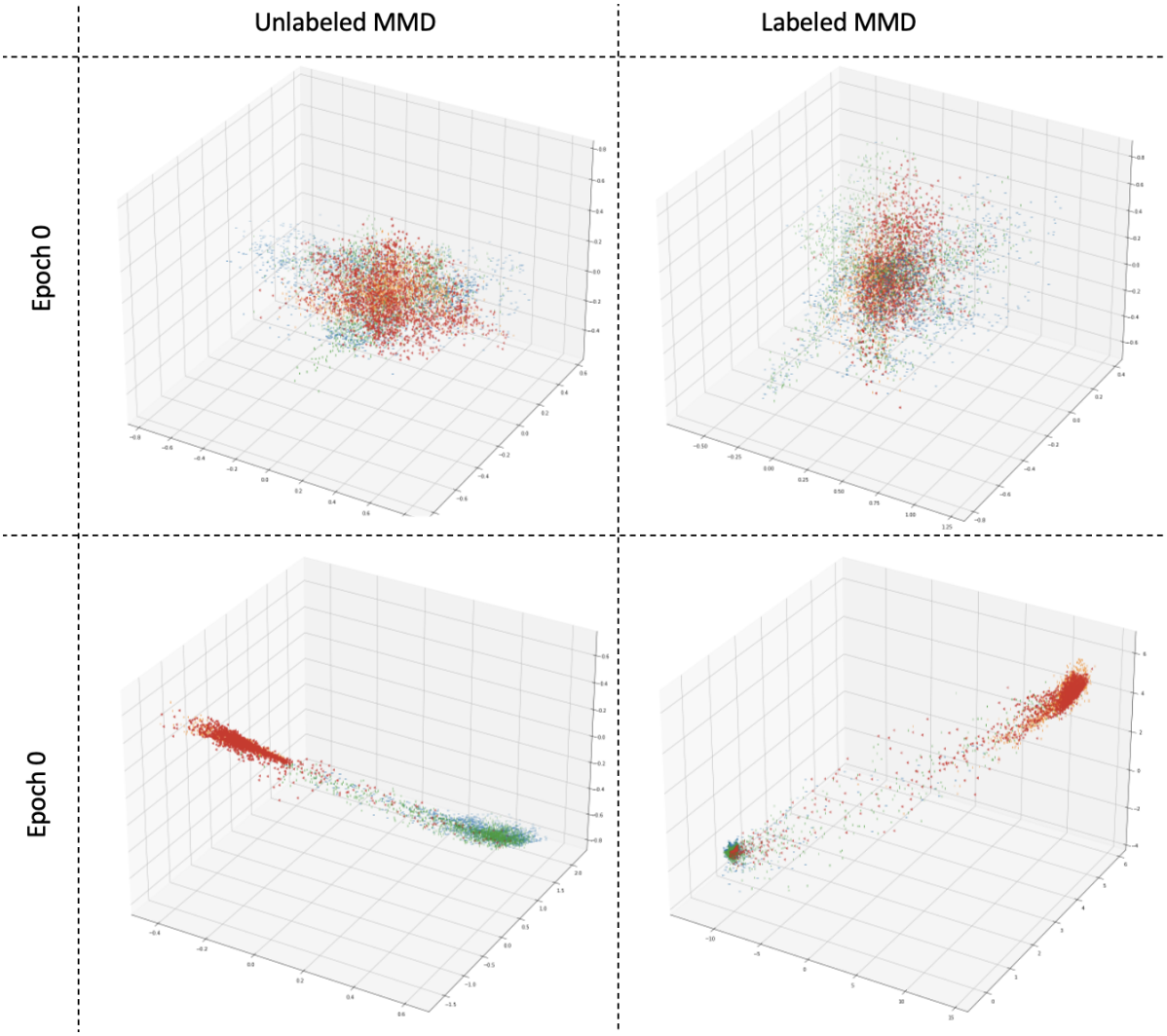

Figure 3.9.: Data distribution: Differences model training with unlabeled vs. labeled MMD loss

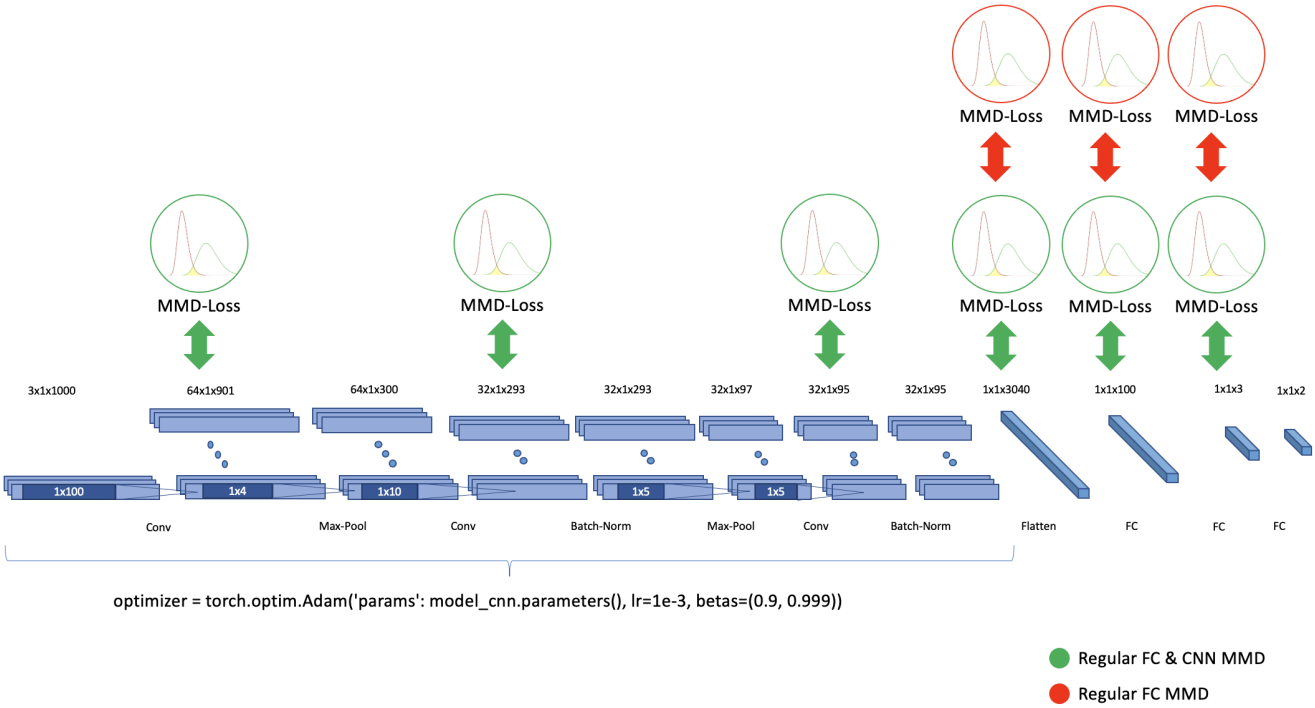

Figure 3.10.: Visualization: Extraction of different latent feature spaces and including them in the MMD loss

```
CNN(
  (conv1): Conv1d(1, 64, kernel_size=(100,), stride=(1,))
  (pool1): MaxPool1d(kernel_size=4, stride=3, padding=0, dilation=1, ceil_mode=False)
  (conv2): Conv1d(64, 32, kernel_size=(10,), stride=(1,), padding=(1,))
  (batch1): BatchNorm1d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (pool2): MaxPool1d(kernel_size=5, stride=3, padding=0, dilation=1, ceil_mode=False)
  (conv3): Conv1d(32, 16, kernel_size=(5,), stride=(1,), padding=(1,))
  (batch2): BatchNorm1d(16, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (fc1): Linear(in_features=1520, out_features=50, bias=True)
)
Classifier(
  (fc2): Linear(in_features=50, out_features=3, bias=True)
  (fc3): Linear(in_features=3, out_features=2, bias=True)
)
```

Figure 3.11.: Model architecture used for evaluating the effect of including different classifier and CNN latent feature spaces in the MMD loss

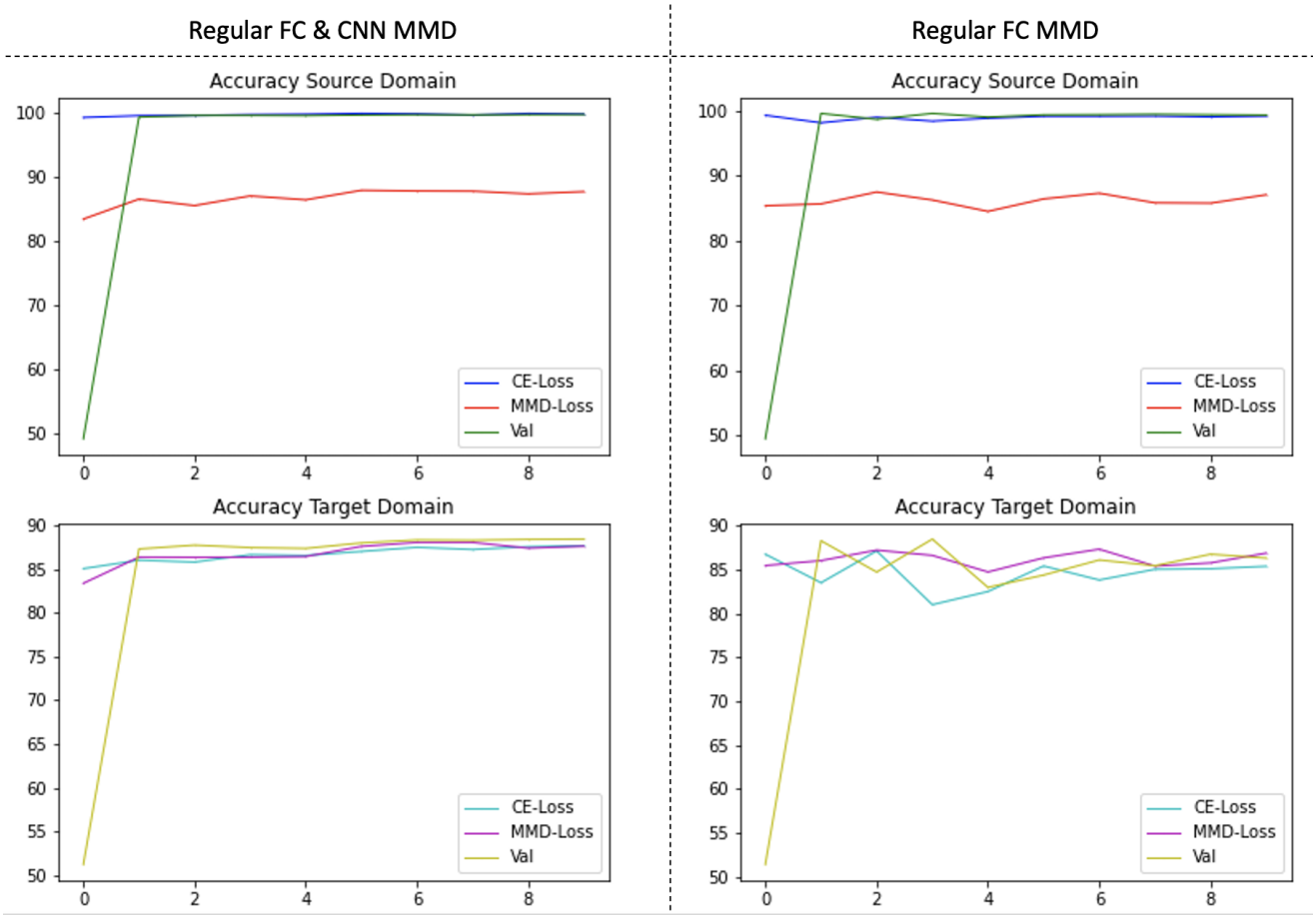

Figure 3.12.: Accuracy curves to evaluate the effect of including different classifier and CNN latent feature spaces in the MMD loss

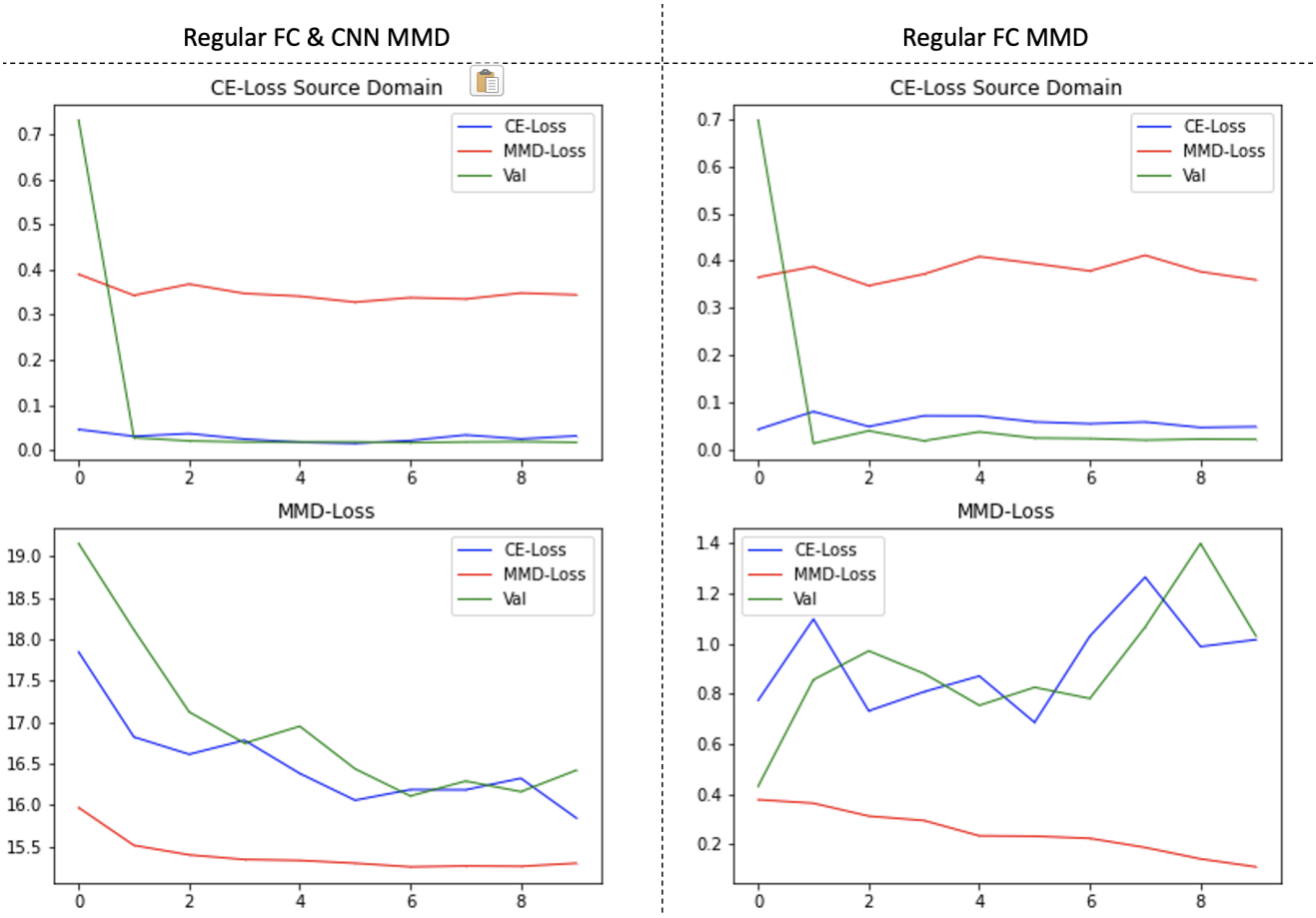

Figure 3.13.: Loss curves to evaluate the effect of including different classifier and CNN latent feature spaces in the MMD loss

3.1.4. Performance of proposed model on Predictive Maintenance approach

In the following section the performance of different MMD-based domain adaption approaches are evaluated on the real dataset.

Model for for iwb Dataset

The model used for the real dataset is close to the one used in the dummy example. One difference is that the architecture just used two max-pooling layers after the first and second convolutional layer. Dataset and the increased number of input features require a more complex model. Three data sequences ('C:x$_b ottom'$,$' C : y_b ottom'$,$' C : z_b ottom')of length 1024 are fed into the CNN. In the classifier

```
CNN(
  (conv1): Conv1d(3, 64, kernel_size=(100,), stride=(1,))
  (pool1): MaxPool1d(kernel_size=4, stride=3, padding=0, dilation=1, ceil_mode=False)
  (conv2): Conv1d(64, 32, kernel_size=(10,), stride=(1,), padding=(1,))
  (batch1): BatchNorm1d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (pool2): MaxPool1d(kernel_size=4, stride=3, padding=0, dilation=1, ceil_mode=False)
  (conv3): Conv1d(32, 32, kernel_size=(5,), stride=(1,), padding=(1,))
  (batch2): BatchNorm1d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (fc1): Linear(in_features=9568, out_features=50, bias=True)
)
Classifier(
  (fc2): Linear(in_features=50, out_features=3, bias=True)
  (fc3): Linear(in_features=3, out_features=2, bias=True)
)
```

Figure 3.14.: MMD Model for real data

On the real machine dataset the three approaches unlabeled, labeled MMD and no MMD loss were evaluated. The accuracies on the source and target domain are visualized in fig. 3.15. The legend of the figure separates between CE-Loss, MMD-Loss and Val. The reason is that the model was trained with 2 optimizer. in total the datset was split in three parts. 60% is used for optimizing the model with an mixed MMD- and cross entropy loss, 20% is used to just train the classifier and 20% is used to validate the model. There is no test data used to evaluate the model. For all three losses the model was trained until convergence of the source data. When training a lot longer than that the performance of the model on the

It becomes obvious that the accuracies on the target domain were abled to be increased with about 10%. The regular unlabeled MMD loss does not use target labels and the labeled MMD loss does not seem to improve the models performance by a lot. Since there is no high performance gains by using the target labels, the focus in the following lies in studying the effectiveness of the unlabeled MMD loss for the problem.

In fig. 3.16 the development of the source domain cross entropy MMD loss is shown. It can be seen, that the hyperparameter GAMMA was picked well, such that the MMD as well as the source cross entropy loss were able to be reduced smoothly throughout the trainings process.

Unfortunately the MMD loss could just minimize the domain discrepancy by a little. The

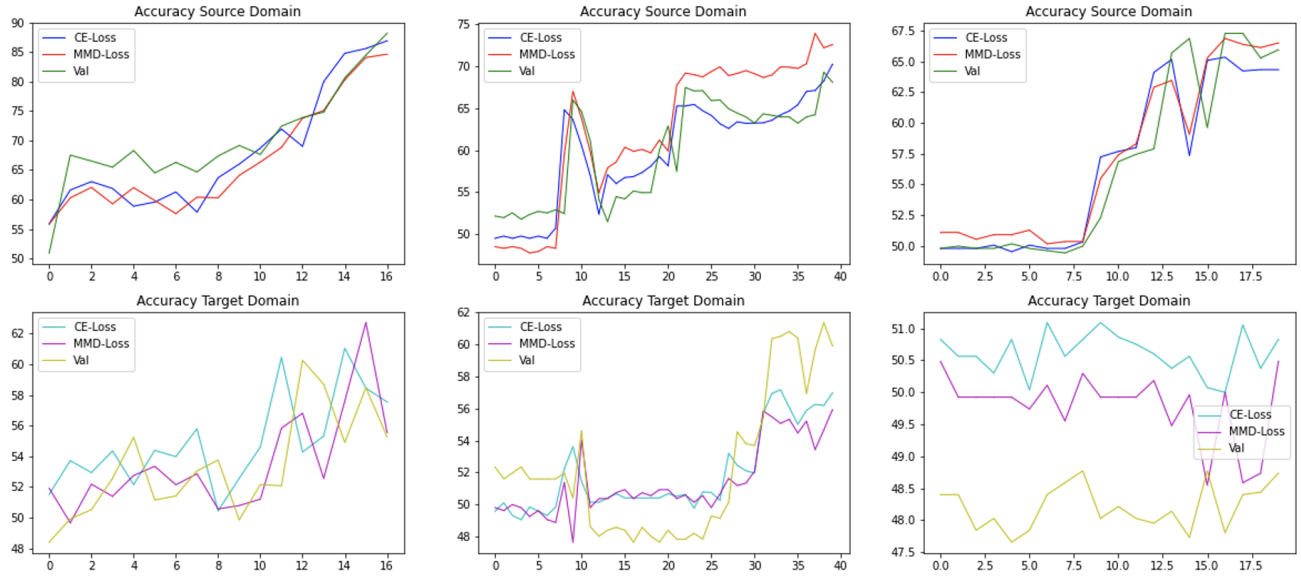

Figure 3.15.: Accuracies for model training with unlabeled MMD loss (left), labeled MMD loss (middle) and no MMD loss (right)

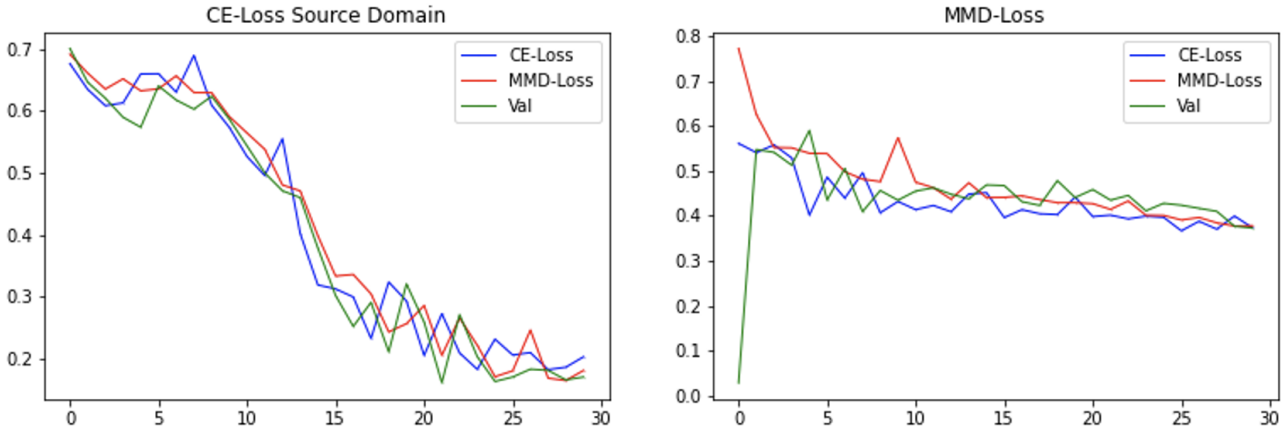

Figure 3.16.: Source domain cross entropy loss and regular unlabeled MMD loss for model training on real machine data

domain discrepancy problem couldn't be solved completely. Still the idea of the MMD loss becomes more clear in the experiments. Also the positive effect of the MMD loss for the training is obvious. For the complex multi-dimensional dataset the MMD loss is probably not effective enough

A. General Addenda

If there are several additions you want to add, but they do not fit into the thesis itself, they belong here.

A.1. Detailed Addition

Even sections are possible, but usually only used for several elements in, e.g. tables, images, etc.

B. Figures

B.1. Example 1

✓

B.2. Example 2

✗

List of Figures

List of Tables

Bibliography

[1] R. Zhao, R. Yan, Z. Chen, K. Mao, P. Wang, and R. X. Gao. "Deep learning and its applications to machine health monitoring". In: *Mechanical Systems and Signal Processing* 115 (2019), pp. 213–237. ISSN: 0888-3270. DOI: https://doi.org/10.1016/j.ymssp.2018.05.050. URL: https://www.sciencedirect.com/science/article/pii/S0888327018303108.

[2] M. Azamfar, X. Li, and J. Lee. "Intelligent ball screw fault diagnosis using a deep domain adaptation methodology". In: *Mechanism and Machine Theory* 151 (2020), p. 103932. ISSN: 0094-114X. DOI: https://doi.org/10.1016/j.mechmachtheory.2020.103932. URL: https://www.sciencedirect.com/science/article/pii/S0094114X20301531.

[3] "Generalization of deep neural network for bearing fault diagnosis under different working conditions using multiple kernel method". In: *Neurocomputing* 352 (2019), pp. 42–53. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2019.04.010. URL: https://www.sciencedirect.com/science/article/pii/S0925231219305247.

[4] V. Pandhare, X. Li, M. Miller, X. Jia, and J. Lee. "Intelligent Diagnostics for Ball Screw Fault Through Indirect Sensing Using Deep Domain Adaptation". In: *IEEE Transactions on Instrumentation and Measurement* 70 (2021), pp. 1–11. DOI: 10.1109/TIM.2020.3043512.

[5] L. Lamport. *LaTeX : A Documentation Preparation System User's Guide and Reference Manual*. Addison-Wesley Professional, 1994.

[6] S. Li, C. Liu, Q. Lin, B. Xie, Z. Ding, G. Huang, and J. Tang. "Domain conditioned adaptation network". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 07. 2020, pp. 11386–11393.

[7] Y. Li, K. Swersky, and R. Zemel. "Generative Moment Matching Networks". In: *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*. ICML'15. Lille, France: JMLR.org, 2015, pp. 1718–1727.

[8] J. Wu, Z. Zhao, C. Sun, R. Yan, and X. Chen. "Few-shot transfer learning for intelligent fault diagnosis of machine". In: *Measurement* 166 (2020), p. 108202. ISSN: 0263-2241. DOI: https://doi.org/10.1016/j.measurement.2020.108202. URL: https://www.sciencedirect.com/science/article/pii/S0263224120307405.