

DATA EFFICIENT DEEP LEARNING FOR ROLLING ELEMENT BEARING FAULT DIAGNOSIS

Mert Sehri

A thesis submitted in partial fulfillment of the requirements for the

DOCTORATE IN PHILOSOPHY

degree in Mechanical Engineering

Ottawa-Carleton Institute for Mechanical and Aerospace Engineering

Faculty of Engineering

University of Ottawa

© Mert Sehri, Ottawa, Canada, 2026

Abstract

Roller element bearings are responsible for 50 to 60 percent of rotating machinery failures, making reliable fault diagnosis critical for industrial machinery health monitoring. This thesis demonstrates that state of the art performance in bearing fault diagnosis does not require massive datasets, but rather well-designed data collection, preprocessing, and loading methods. Focusing on the entire data lifecycle from acquisition to model input, this work challenges the notion that extensive data is required to train machine learning (ML) models for condition monitoring by showing that carefully structured and efficiently loaded data can deliver superior cross domain generalization even with limited samples. Benchmark publicly available bearing datasets are analyzed, revealing weak cross domain coverage that restrict the transferability of ML models to industrial scenarios. To address this gap, a new bearing dataset design is proposed under controlled conditions with naturally developed faults, and a novel selective embedding data loading method is introduced, which alternates sensor inputs in a structured and alternating fashion to maximize training diversity and prevent overfitting without increasing data volume. Theoretical foundations connecting selective embedding to gradient variance reduction, effective sample size, and generalization bounds are established to explain why the method works. Cross domain experiments are conducted across heavy machinery, manufacturing, and railway datasets to confirm that selective embedding can achieve over 90% classification accuracy under challenging domain splits, outperforming both single-channel and parallel loading methods that require

significantly more data or computational effort. These findings establish that data efficiency, not data size, is the key to scalable, transferable, and computationally affordable deep learning for industrial bearing fault diagnosis.

Acknowledgement

I would like to sincerely thank my supervisor, Dr. Patrick Dumond, for his guidance, patience and support throughout this journey. His mentorship has formed not only this thesis but also my development as a researcher and I am very thankful for the opportunities and trust he has given me over the years.

I would like to thank my father Dr. Kaan Sehri for his support for my education and for helping me start research. I am also thankful to my mother, Derya Sehri, for her emotional and overall support during this journey. I also want to thank my sister, Mina Sehri, for reading my work and giving me valuable feedback throughout the process.

I would like to thank my collaborators and friends, Francisco, Flavio, Merve, Govind, Tongtong and Ralf, for many valuable discussions on my research topics and their support during this process.

Last but not least, I wish to thank everyone else who contributed to this work, directly or indirectly, including colleagues and the broader research community, whose previous work has made this thesis possible.

Table of Contents

Table of Contents	v
List of Figures.....	viii
List of Tables.....	x
Nomenclature	xi
Chapter 1 Introduction.....	1
1.1 Motivation	4
1.2 Strategy	5
1.3 Expected Thesis Contributions	6
1.4 Thesis Outline	7
Chapter 2 Background.....	9
2.1 Bearings.....	9
2.1.1 Bearing Fault Frequencies.....	11
2.1.2 Bearing Failure Progression	20
2.1.3 Bearing Life Analysis.....	21
2.2 ML for Fault Diagnosis	22
2.2.1 Traditional versus Deep Learning	22
2.2.1.1 Accuracy.....	24
2.2.1.2 Hyperparameters.....	25
2.2.2 Data Loading Methods.....	26
Chapter 3 Methodology.....	29
3.1 Datasets.....	29
3.1.1 Benchmark Bearing Datasets	30
3.1.1.1 CWRU Dataset	30
3.1.1.2 Triaxial Bearing Dataset.....	31
3.1.2 Data Loading Evaluation Datasets.....	32

3.1.2.1	Industrial Steel Slag Flow Dataset.....	33
3.1.2.2	UOEMD Dataset.....	33
3.1.2.3	IEEE PHM Beijing 2024 Data Challenge Dataset	34
3.2	Signal Preprocessing	36
3.2.1	Min-Max	37
3.2.2	Z-score.....	37
3.2.3	RMS	38
3.2.4	Fast Fourier Transform (FFT)	38
3.3	ML Architectures.....	40
3.3.1	Benchmark Bearing ML Architectures.....	40
3.3.2	Data Loading Evaluation ML Architectures	43
3.4	Data Domain Splitting	44
Chapter 4	Dataset Design and Analysis	51
4.1	Proposed Dataset Design	51
4.1.1	Experimental Setup	53
4.1.2	Data Acquisition System.....	56
4.1.3	Data Collection	57
4.1.4	Raw Dataset Design.....	59
4.2	Bearing Dataset Analysis.....	63
4.2.1	CWRU Analysis.....	63
4.2.2	Triaxial Bearing Analysis	71
4.2.3	UORED-VAFCLS Analysis	74
4.2.4	Discussion.....	76
Chapter 5	Data Loading Methods.....	81
5.1	Traditional Data Loading.....	82
5.2	Selective Embedding	83

5.2.1	Method Overview	84
5.2.2	Experiment Setup	91
5.2.3	Results and Discussion	98
Chapter 6	Theoretical Foundations of Selective Embedding.....	106
6.1	Problem Setup (Defining the Data and Domain)	107
6.2	Redundancy Versus Diversity.....	108
6.3	What Selective Embedding Changes	109
6.4	Theoretical Information	110
6.5	Benefits of Selective Embedding.....	111
6.5.1	Effective Sample Size.....	111
6.5.2	Gradient Variance Reduction	112
6.5.3	Uniform Mixture Approximation.....	112
6.5.4	Generalization via Stability.....	113
6.5.5	Risk Decomposition.....	114
6.5.6	Practical Takeaway.....	116
Chapter 7	Conclusion.....	117
7.1	Summary of Completed Work.....	117
7.2	Future Work	119
References		120

List of Figures

Figure 2-1. a) Contact angle [12], b) Rolling element component identification [13], c) Important dimensions of the rolling element [14]	13
Figure 2-2. Frequency vs. bore diameter for SKF bearing inner race frequencies with various motor speeds, 10 mm to 200 mm.....	18
Figure 2-3. Stages of bearing failure [21]	21
Figure 2-4. Existing data loading methods [26]	28
Figure 3-1. CWRU Bearing Data Test Rig [30]	31
Figure 3-2. Triaxial Bearing Data Test Rig [27].....	32
Figure 3-3. Industrial Data Collection Setup [32]	33
Figure 3-4. UOEMD Test Rig Setup [34]	34
Figure 3-5. IEEE PHM Beijing 2024 Data Challenge Test Rig Setup [35].....	35
Figure 3-6. CWRU Dataset Splitting [29]	47
Figure 3-7. Triaxial Bearing Dataset Splitting	48
Figure 3-8. Domain loading of a) Dataset 1- Steel slag flow, b) Dataset 5- UOEMD, c) Dataset 6- IEEE PHM Beijing 2024 data challenge.	50
Figure 3-9. Domain loading of Dataset 4- UORED-VAFCLS.....	50
Figure 4-1. UORED-VAFCLS Test Rig Set Up in Operation a) Front View b) Side View	55
Figure 4-2. a) Two-Toothed Gear, b) Accelerometer Placement (Front View Close-up) [39].	55
Figure 4-3. Dataset Design Block Diagram	61
Figure 4-4. Accelerometer Data for a) O-7-2 and b) H-7-0	62
Figure 4-5. Acoustic Data for a) O-7-2 and b) H-7-0.....	63
Figure 4-6. FFT Validation for a) O-7-2 and b) H-7-0	63

Figure 4-7. Average Test Accuracy When Splitting Data by Fault Severities with the CWRU Dataset.....	70
Figure 5-1. Comparison of Proposed Data Loading Method versus Existing	84
Figure 5-2. Selective Embedding Input Data Loading Across Different Sources	85
Figure 5-3. Comparison of input data loading methods: (a) traditional single channel, (b) traditional parallel multi-channel, and (c) proposed selective embedding with alternating sources.	90
Figure 5-4. CWRU dataset domains showing time- and frequency-domain signals with varying fault severities.....	94
Figure 5-5. Selective embedding with synchronized accelerometer and acoustic signals from the UORED-VAFCLS dataset [26].....	96
Figure 5-6. Test accuracy distribution for data loading methods for all datasets using a simple CNN architecture [26].....	102
Figure 5-7. Test accuracy distribution for data loading methods for all datasets using a CNN-Transformer [26].....	103
Figure 5-8. Radar plot using datasets 1, 5, and 6 for different ML algorithms [26].....	104
Figure 5-9. Radar plot using datasets 2, 3, and 4 for different ML algorithms [26].....	105

List of Tables

Table 3-1 Datasets selected for this thesis	35
Table 3-2 Network architectures of the methods compared.....	44
Table 3-3 Domain Splitting for the CWRU Dataset 48 kHz Drive End.....	46
Table 4-1 Bearing Specifications.....	54
Table 4-2 Dataset Numbering.....	61
Table 4-3 Accelerometer Results of Different ML Models for the CWRU Dataset.....	66
Table 4-4 Accelerometer Results for Existing Fault Severity Domains of the CWRU Dataset	69
Table 4-5 Accelerometer Results of Different ML Models for the Triaxial Bearing Dataset	73
Table 4-6 Accelerometer Results of Different ML Models for the UORED-VAFCLS Bearing Dataset.....	75
Table 5-1 Algorithm 1: pseudocode for selective embedding.	91
Table 5-2 Comparison of data loading methods when using a CNN and a CNN-Transformer	102

Nomenclature

ACDIN	Atrous Convolutional Deep Inception Network
BPFO	Ball Pass Frequency of Outer Race
BPFI	Ball Pass Frequency of Inner Race
BSF	Ball Spin Frequency
CNN	Convolutional Neural Networks
CWRU	Case Western Reserve University
DANN	Domain-Adversarial Neural Network
DE	Drive End
DFT	Discrete Fourier Transform
DL	Deep Learning
FE	Fan End
FFT	Fast Fourier Transform
FTF	Fundamental Train Frequency
IIoT	Industrial Internet of Things
KNN	k-Nearest Neighbour
ML	Machine Learning
MMD	Maximum Mean Discrepancy
RMS	Root Mean Square
RNN	Recurrent Neural Network
SGD	Stochastic Gradient Descent
SVM	Support Vector Machine

STFT	Short Time Fourier Transform
TL	Traditional Learning
UOEMD	University of Ottawa Electric Motor Dataset
UORED-VAFCLS	University of Ottawa Rolling-element Dataset – Vibration and Acoustic Faults under Constant Load and Speed conditions
WDCNN	Wide First-Layer Kernel Deep Convolutional Neural Network

Chapter 1

Introduction

Machine components are critical in heavy machinery, allowing machines to transfer heavy loads under harsh conditions. Specifically, roller element bearings reduce friction between stationary and rotating elements during this load transfer. Despite the significance of individual machine components, roller element bearings are responsible for 50% to 60% of rotating machinery failures, making them the leading cause of failure [1], [2]. The high failure rate of bearings underscores the importance of bearing health data collection for conducting fault diagnosis and prognosis. Diagnosis refers to the identification of a fault condition using commonly known symptoms, while prognosis focuses on predicting the outcome of a condition. Furthermore, prognosis plays a crucial role in determining the timing of machine maintenance.

In modern systems, the health condition of machinery can be determined using component data. For both diagnosis and prognosis, large amounts of data need to be collected, capturing measurements from healthy to failure states. Historically, this data has been used to diagnose faults using simple methods (e.g., the Fast Fourier Transform (FFT) and characteristic fault frequencies).

In recent years, maintenance engineers have moved away from traditional vibration analysis methods and towards intelligent machine learning (ML) methods. This is because domain experts are becoming harder to come by while machines and maintenance methods are becoming more complicated. Traditional methods work to some extent but depend too much on fixed thresholds and pre-defined patterns. These don't always account for all possible problems that can occur in modern, complex machinery systems. Also, these methods usually need someone with domain expertise that can interpret collected data by hand, which can take a lot of time and is prone to mistakes. Unfortunately, there are fewer domain experts available even though the need for them is growing. On the other hand, ML methods have made maintenance processes more data-driven, creating the possibility for real-time analytics and predictive capabilities. In particular, when combined with Industrial Internet of Things (IIoT), ML allows for continuous monitoring of bearings and early fault detection. This improves the reliability and efficiency of machine maintenance. However, this new data-driven approach needs more flexible, scalable and smart solutions to manage the large quantities of data created by modern heavy machinery.

Current ML algorithms have generally been designed with lab-generated data, which are not representative of real-world environments. Thus, ML algorithms almost always fail to function in practical applications. For this reason, intelligent machine condition monitoring has yet to see wide-scale adoption.

The purpose of this thesis is to clarify the main features related to data collection parameters influencing the training of ML algorithms and to investigate data pre-

processing and loading methods that can help improve ML-based classification methods. Other than developing an understanding of how ML algorithms work with vibration and acoustic data, to achieve better effectiveness of bearing diagnosis, one must also simplify the data collection process. Understanding how ML models interpret data and detect bearing states is the first step to closing the gap between research and practical applications. This comprehension will allow the optimization of algorithm accuracy and efficiency, leading to a practical solution for bearing fault diagnosis. Additionally, simplifications to the data collection process when using ML methods will be explored so that hardware and computational requirements can be reduced, another key limitation of current IIoT systems.

Addressing IIoT limitations requires a clear understanding of ML methods available for bearing fault diagnosis. ML can be sub-divided into two branches: traditional learning (TL) and deep learning (DL), both of which are used in bearing diagnosis and prognosis. However, DL methods have gained in popularity since LeCun et. al's DL paper in 2015 [3]. To integrate bearing fault detection with ML bearing diagnosis, state of the art algorithms must be combined with methods for simplifying the number of samples collected and reducing hardware and computational requirements. This will serve as a first step toward translating research into practical applications.

The need for effective data collection is the basis for exploring the interaction between bearing datasets and ML algorithms. The understanding of these interactions is crucial as it is a prerequisite for developing effective and accurate DL

models for machinery fault diagnosis. The interaction between data and algorithms is crucial, as the success of ML algorithms depends largely on the availability of high-quality and diverse data to learn the underlying patterns, detect faults, and make accurate predictions. ML performance is dependent on the design of the datasets used, relevant operating condition features, and the way that various challenges such as noise, missing data, and class imbalance are addressed. Therefore, the absence of bearing datasets that are designed with ML-specific controlled data collection practices must be addressed. These include missing data, and class imbalance that limit the development of robust and generalizable ML models capable of diagnosing faults across a variety of machinery conditions and environments.

1.1 Motivation

There are numerous research papers indicating the need for understanding data collection specifically for ML-based fault diagnosis [4], [5]. However, existing bearing datasets have rarely considered ML in their design, nor have they provided a justification for the data collection and analysis methods used [6], [7], [8]. Furthermore, there are still significant gaps, including a limited understanding of which architectures to use when implementing ML classification problems. Finally, there is a lack of clarity around the use of data preprocessing methods and input data loading methods, especially related to their effect on ML algorithm performance [9]. Therefore, this thesis will address these issues by focusing, not only on how data should be collected and designed, but also on how it is preprocessed, structured, and introduced into ML models during training and testing to reach high test accuracies

for fault diagnosis. By optimizing end-to-end data processing, from acquisition to input representation, this thesis aims to enhance understanding of how to effectively use ML fault diagnosis, leading to more reliable condition monitoring systems capable of accurately diagnosing machinery faults under various conditions.

1.2 Strategy

This thesis proposes a data loading method for rolling element-bearings to overcome these limitations and enhance the application of intelligent fault diagnosis. The aim of this research is to improve the state of the art in DL-based machine condition monitoring by investigating the collection of bearing data, the use of this data by ML algorithms, and to provide more robust and reliable tools for fault diagnosis.

The high-level objectives of this thesis are to:

- Evaluate existing benchmark bearing datasets to identify the highest cross domain test accuracy possible and to pinpoint which input signal lengths and preprocessing methods to be used for fault diagnosis applications. This will improve cross domain generalizability, contributing to more transferable diagnostic models.
- Develop a new dataset for ML fault diagnosis that connects bearing signature theory to ML-based fault diagnosis to achieve high condition monitoring fault diagnosis test accuracies.
- Assess ML algorithms, data loading methods and input types (e.g., sensors) to improve model performance and generalization.

1.3 Expected Thesis Contributions

This thesis contributes to the field of DL-based machine condition monitoring, including both data collection and optimization of ML algorithms. Motivated by the limitations of existing bearing datasets and algorithm implementations, key contributions of this thesis include:

- Identifying the relationship between bearing data and data preprocessing methods, including how different data input methods affect model generalization and diagnostic accuracy across domains. This is achieved by doing a benchmark analysis of existing publicly available bearing datasets, identifying their cross domain limitations and determining the input signal length and preprocessing method that yield the highest diagnostic accuracy.
- Developing a new bearing dataset as a first step toward mimicking a practical setup in a controlled lab setting, enhancing ML algorithm training for cross domain bearing fault diagnosis. This is followed by an analysis of the dataset by looking at data structure and loading to improve performance of ML-based condition monitoring fault diagnosis.
- Proposing selective embedding as a novel data loading method that alternates between data sources within a single channel input to increase training diversity without increasing data volume, achieving improved cross domain generalization compared to traditional single channel and parallel loading methods.

These contributions aim to advance the field by providing a more robust, versatile, and realistic ML-based condition monitoring approach for bearing data collection and use, addressing identified limitations and enhancing the effectiveness and applicability of DL models in practical industrial fault diagnosis applications.

1.4 Thesis Outline

Chapter 2 lays the groundwork for the mechanics and ML that this work is based on. It discusses the theory of rolling element bearings, such as fault frequencies, failure progression, and life analysis. It also connects these ideas to fault diagnosis. Furthermore, the chapter discusses the differences between traditional and DL methods and how important data loading methods are for getting strong generalization.

Chapter 3 looks at current ML methods using publicly available bearing datasets. This chapter also looks at how common preprocessing methods (min-max, Z-score, RMS, and FFT) change the quality of the data and the stability of the model. Finally, this chapter talks about the different ML architectures used in this work and sets up the next chapter for benchmark cross domain evaluation.

Chapter 4 shows the results of cross domain experiments using publicly available datasets. It shows how preprocessing and domain splitting methods affect diagnostic performance. The results show that FFT preprocessing always helps generalization and that realistic domain separation is important for avoiding data leakage and false high accuracy results. These results inform the design of a new bearing dataset that aims to fix limitations that exist with current datasets. This new dataset has longer

data collection durations, controlled operating conditions, and clear labelling, which make it possible to repeat experiments and supports both TL and DL methods.

In Chapter 5 the selective embedding data loading method is proposed. This method switches between sensor input segments to make training more diverse without needing more data. The chapter compares the CNN-LSTM and other ML architectures using this new loading method. It shows that selective embedding achieves more than 90% accuracy on datasets obtained from heavy machinery, railroads, and manufacturing. These findings validate that data-efficient training methodologies can either match or exceed those requiring substantial amounts of data, in accordance with the thesis objective of transcending the limited data problem.

Chapter 6 provides the theoretical foundation for selective embedding. This is followed by a discussion of how diversity and generalization are affected by selective embedding. The chapter concludes with the benefits of selective embedding, including a reduction in the required sample size and other practical takeaways.

Finally, Chapter 7 provides concluding remarks and a discussion on future work.

Chapter 2

Background

2.1 Bearings

Rolling element bearings are fundamental components of rotating machines to allow smooth rotation of shafts carrying radial and axial loads. Their main purpose is to reduce friction between moving and non-moving parts, so that power can be transferred efficiently without parts wearing out too quickly. The four main parts of a rolling element bearing are the inner race, outer race, rolling elements (balls or rollers), and cage that holds the rolling elements at equal distances from each other. Each of these components plays a role in how forces are transferred and stresses accounted for while the system is running.

In mechanical design, parameters like bore diameter, load capacity, material hardness and lubrication conditions determine the choice of a bearing. Hertzian contact theory describes the mechanics of contact within a bearing [10]. The theory states that high stresses are concentrated at the contact points between the rolling elements and the races. Such stresses often determine the life of the bearing prior to failure. With more load, subsurface stresses can cause cracks to form in places. Given

time these cracks can develop into spalls and pits on the rolling elements or raceways. This process is the reason why things go wrong in industrial settings.

In mechanical systems, bearings often operate under variable load and speed conditions. These variations affect the vibration response, making fault detection more complex in real-world environments when compared to controlled laboratory test rigs. While seeded fault datasets often provide clean signals for benchmarking, they fail to capture the complexity of naturally developed defects, where signal amplitudes are modulated by load paths, effects of resonance, and system-level dynamics. This mismatch between lab-generated and industrial data has been one of the main challenges in transferring ML algorithms into practice.

Another important aspect of bearing operation is lubrication. Proper lubrication forms a film that minimizes the direct metal to metal contact and reduces friction and wear. Lack of lubrication or contamination, however, accelerates failure initiation due to higher surface stresses and formation of local hotspots. The vibration response is also complicated by uneven stress distributions in the bearing due to misalignments or unbalanced loads. These mechanical factors justify the need of condition monitoring for both signal analysis and ML methods able to handle non-linear, time-varying data.

This thesis connects bearing signature theory and data-driven methods using basic engineering concepts. This background ensures that the creation of datasets and ML models is based on the mechanical behaviour of bearings and not only on algorithmic assumptions.

2.1.1 Bearing Fault Frequencies

A key component of data collection is understanding the signal of interest. Bearing test rigs often focus on creating signals that capture fundamental bearing fault frequencies since controlled research lab environments allow for relatively clean data collection with minimal interference. In such cases, high-frequency resonant bands, which are typically much higher than the fundamental fault frequencies, can sometimes be overlooked. Unfortunately, in industry, fundamental bearing fault frequencies are often buried in machine and environmental noise. Therefore, early detection often relies on more distinct high-frequency signal content. As highlighted by Randall and Antoni [11], bearing faults often excite high-frequency structural resonances, which depend more on the machine setup, such as the test rig, rather than the bearing type itself. However, in lab test rigs, data is generally less noisy, allowing researchers to focus on the fundamental bearing frequencies with fewer concerns for high-frequency resonance interference.

Moving to real-world industrial environments introduces significant operational noise, making it crucial to consider these high-frequency resonant bands. In such cases, high-frequency bursts caused by impacts due to faults excite resonant frequencies, and these signals are further modulated by the load and transmission path. Ignoring these high-frequency components can result in improper fault detection during traditional analysis, as the fundamental fault frequencies and their harmonics can easily be masked by noise from other machinery. Methods like envelope analysis and spectral kurtosis become essential for capturing these high-

frequency signals, which carry significant diagnostic information in noisy industrial environments. While lab test rigs might allow for a focus on fundamental frequencies, ensuring that the frequency range in industry data collection includes these high-frequency resonant bands is crucial for effective traditional fault diagnosis. However, due to upper frequency limitations in existing IIoT devices, as well as the lack of publicly available industrial datasets, this research seeks to understand how ML algorithms interact with bearing fundamental frequencies to provide fault diagnosis. It is hoped that ML algorithms will provide additional capacity to filter through noisy industry data rather than require high-frequency resonant bands to provide adequate performance.

To better understand rolling element bearings, Figure 2-1 a), b), and c) are provided to visualize their characteristic features. In most cases, researchers are interested in classifying bearing fault types. Bearing faults are generally categorized into four main types: inner race faults, outer race faults, ball (rolling element) faults, and cage faults. Each produces characteristic vibration signatures due to repetitive impacts during operation. Of the four types of faults, the inner race is usually the most common location for failure. This is because the inner race has a smaller diameter and rotates with the shaft. Therefore, it has more load cycles per revolution than the larger stationary outer race, and higher contact stresses due to a smaller radius of curvature. Failures involving the ball and cage are less common, with damage to the ball often being a secondary consequence of race spalling rather than an independent origin of failure, whereas cage failures are generally limited to

situations of severe misalignment or absence of lubrication. These characteristic defect frequencies include the Ball Pass Frequency Outer (BPFO) race frequency, the Ball Pass Frequency Inner (BPFI) race frequency, the Ball Spin Frequency (BSF), and the Fundamental Train Frequency (FTF), all of which are calculated from geometric bearing parameters such as the ball diameter, pitch diameter, number of rolling elements, and contact angle. Because these frequencies are tied directly to bearing geometry and shaft rotational speeds, they are used extensively for both traditional vibration analysis and modern ML condition monitoring.

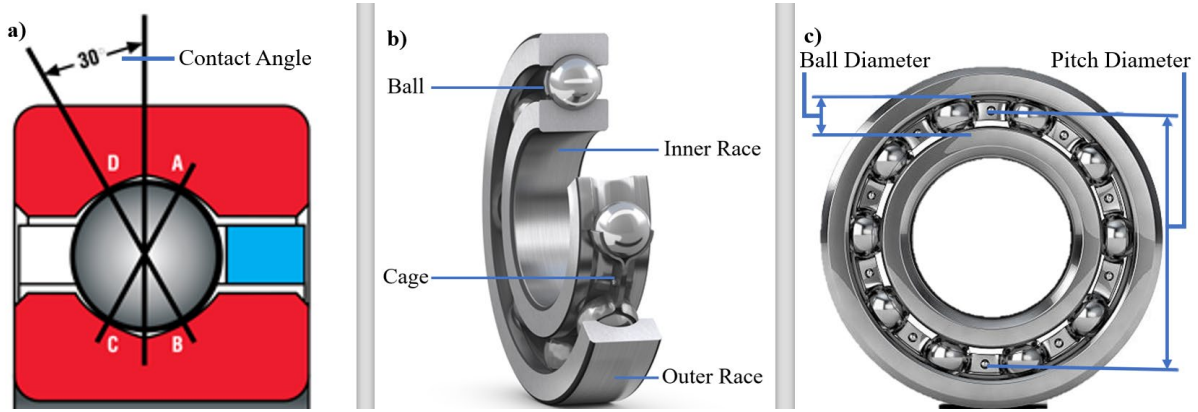


Figure 2-1. a) Contact angle [12], b) Rolling element component identification [13],
c) Important dimensions of the rolling element [14]

Rolling element bearing fault frequencies play an important role in identifying these fault types. Characteristic bearing fault frequency equations can be written as:

$$BPFO = \frac{nf_r}{2} \left(1 - \frac{d}{D} \cos\phi\right) \quad (1)$$

$$BPFI = \frac{nf_r}{2} \left(1 + \frac{d}{D} \cos\phi\right) \quad (2)$$

$$FTF = \frac{f_r}{2} \left(1 - \frac{d}{D} \cos\phi\right) \quad (3)$$

$$BSF = \frac{f_r D}{2d} [1 - (\frac{d}{D} \cos \phi)^2] \quad (4)$$

where D represents the pitch diameter of the bearing, d is the rolling element (ball) diameter, n is the number of rolling elements and ϕ is the load contact angle. Physically, each characteristic fault frequency corresponds to the rate at which a defect repeatedly makes contact during rotation. For an inner race fault, when a rolling element passes over a scratch or pit on the inner race, it briefly dips into the defect and then strikes the race again as it exits, producing a sharp impact once per pass. Since the inner race rotates with the shaft, this occurs at the BPFI rate. An outer race fault behaves similarly, but because the outer race has a larger diameter, rolling elements pass over the same defect location at a lower rate (BPFO), producing a comparable dip-and-impact pattern each time a ball passes that fixed point. A ball fault produces an impact each time the defect on the rolling element itself contacts a race, occurring at the ball spin frequency (BSF). Since each ball contacts both the inner and outer race once per spin, a ball defect typically produces two impacts per revolution of the ball. A cage fault disrupts the even spacing of the rolling elements as the cage rotates at the fundamental train frequency (FTF), causing irregular or uneven impact timing between rolling elements and the races rather than a single sharply periodic impact. In each case, these repeated mechanical impacts appear in the frequency domain as sharp harmonic peaks at the corresponding characteristic frequency and its integer multiples, which is why spectral analysis at these specific frequencies provides a diagnostic for both the presence and the location of a fault.

Characteristic bearing frequencies are usually associated with a set of harmonic frequencies in integer intervals of the characteristic frequency. Bearing fault frequency harmonics are known to be useful for understanding the type and severity of the faults contained within a signal. Bearing faults come from the formation of cracks or pits in bearing surfaces that cause impacts, which are reflected in vibration signals as a spike [15]. These faults can also generate harmonics that are primarily driven by non-linear loads or conditions. Since these harmonics play such a key role in identifying differences between similar characteristic bearing fault values, providing enough data to identify these bearing fault harmonics is important. The number of fault harmonics available in a signal depends on the value of the characteristic bearing fault frequency and the sampling rate. The total number of fault harmonics contained in a signal can be calculated as half the sampling rate divided by the highest identified bearing fault frequency, as shown in equation (5).

$$\# \text{ of Fault Harmonics} = \frac{F_s/2}{F_{Hbearing}} \quad (5)$$

In the tutorial paper for “Rolling element bearing diagnostics” provided by Randall and Antoni, the use of 10 or more harmonics in capturing bearing fault frequencies is essential for identifying bearing faults using traditional methods [11]. Thus, within this tutorial framework, 10 fault harmonics serve as a minimum requirement for fault diagnosis by an expert when collecting data without the intended use of ML.

This thesis only focuses on rolling element ball bearings, which are the most common type of bearings used in small to medium sized industrial machinery such

as motors, pumps and fans as discussed in section 2.1.1. Cylindrical roller bearings, chosen for higher load applications, are not considered as part of this work.

To examine the impact of ball bearing fault frequencies on data collection, a range of SKF bearings with varying bore diameters are selected. This selection includes the entire spectrum of commonly available bearings, starting from the smallest bore size and extending to the largest bore diameter available. The goal is to observe how different bearing frequencies (i.e., BPFI, BPFO, BSF, and FTF) affect the data collection process.

The SKF 6000 bearing is selected to be the smallest diameter used frequently in industry. This helps set the frequency range possible [16]. Although smaller bearings are manufactured by SKF and are used for special applications requiring high speeds, they are not often used in industry. The SKF 6000 dimensions include a 26 mm outer diameter, a 10 mm bore diameter, a pitch diameter of 18 mm, and a ball diameter of 6 mm. The bearing contains 9 balls and is rated for a maximum rotation speed of 40,000 RPM. Loads are assumed to be applied only in the radial direction at the maximum motor speed. In this case, characteristic bearing frequencies include a BPFO of 2,166.7 Hz, a BPFI of 3,833.3 Hz, an FTF of 240.7 Hz, and a BSF of 1,107.4 Hz. SKF's largest single-row ball bearing in common use, the 708/1250AMB [17], has an outer diameter of 1500 mm, a bore diameter of 1250 mm, a pitch diameter of 1375 mm, and a ball diameter of 55 mm. It contains 14 balls and is rated for a maximum speed of 280 RPM. Loads are assumed to be applied only in the radial direction at 280 RPM. Based on this information, the largest bearing will have a

BPFO of 31.4 Hz, a BPFI of 34.0 Hz, an FTF of 2.2 Hz, and a BSF of 58.2 Hz. From equations (1)-(4), rotational speed can be seen to have the most significant effect on bearing frequencies for data collection, especially since the number of balls cannot be adjusted due to each manufacturer's specifications.

The fault frequency range of SKF bearings is obtained by calculating the characteristic bearing frequencies in the bore range of 10 to 1250 mm. Tests were only conducted at typical industry speeds to determine the bearing fault with the highest frequency. Figure 2-2 provides a sub-set of results for the highest fault characteristic frequencies obtained at common AC and DC motor speeds [18] used in industry. Since the largest common bearings operate well below 900 RPM and have lower fault frequencies, they are not included in this figure. The inner race is specifically chosen due to it having the highest frequency among the four fault signatures. Figure 2-2 exhibits the highest frequency of 490 Hz at 5000 RPM, representing a single fault harmonic. This frequency aligns reasonably well with industry expectations and can be considered as the typical maximum fault signature frequency.

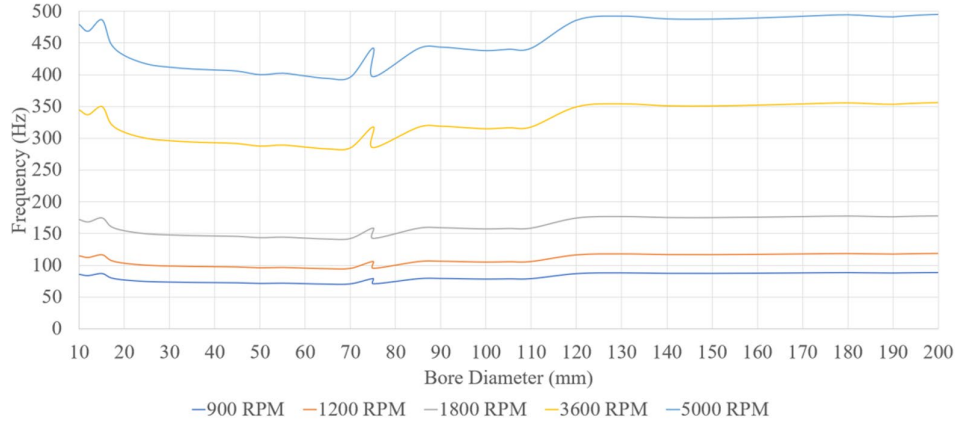


Figure 2-2. Frequency vs. bore diameter for SKF bearing inner race frequencies with various motor speeds, 10 mm to 200 mm

In particular, the 6203 bearing will be closely examined in this thesis as it is the most widely researched rolling element bearing in the condition monitoring community. This is largely due to its use in applications for small to medium-sized electric motors, pumps, and fans, which make up a substantial portion of rotating machinery in industry. Its compact size, standardized dimensions (17 mm bore, 40 mm outer diameter, 12 mm width), and low cost have made it the default choice for laboratory test rigs and benchmark datasets such as the Case Western Reserve University (CWRU) bearing dataset [19]. However, these datasets almost exclusively rely on seeded faults, which do not fully represent the complexity of naturally occurring defects found in real-world machinery. In this thesis, the 6203 bearing will be used not only for its industrial relevance but also as the basis for proposing a new test rig designed to generate datasets that accommodate naturally developed faults. Since no publicly available industrial data exists to capture these natural fault

progressions, this approach provides a critical step in bridging the gap between controlled laboratory studies and actual industrial conditions.

Finally, it is important to remember that fault harmonics play an important role in the diagnosis of bearing faults when using ML algorithms. This means that several integer values higher than the highest characteristic frequency will be required to properly identify faults. A few factors must also be considered: differentiating between different fault types and noise, reliability, and redundancy. Differentiating between different fault types and noise in bearing data is important so that enough fault harmonics are available to classify differences between each class type when trying to obtain high ML algorithm accuracies. Reliability and redundancy refer to gathering enough fault harmonics to have a clear, distinguished signal. This is crucial for effectively differentiating between potential noise in data and bearing fault signatures. The goal is to ensure that there are enough fault harmonics included in the signal to enable the network to recognize and accurately respond to various fault conditions. Reliability involves the consistent and accurate performance of the system in identifying and distinguishing between healthy and faulty conditions. Redundancy, on the other hand, consists of additional components and measures beyond the essential requirements, providing backup functionality to strengthen overall system reliability. In this way, the combination of reliability and redundancy contributes to a robust fault detection mechanism.

2.1.2 Bearing Failure Progression

From a holistic perspective, bearing failure does not occur instantaneously but rather progresses through a series of stages [20]. A simple process illustrates this progression, as shown in Figure 2-3:

Stage 1 is the healthy state, the bearing surfaces are intact, with proper lubrication and no visible damage. Vibration signals remain steady with only baseline noise.

Stage 2 is the crack initiation, localized subsurface stresses, often described by Hertzian contact theory, initiate small cracks below the raceway surface. These cracks are invisible at first but begin to affect the vibration response.

Stage 3 is the spall formation, cracks propagate to the surface, creating spalls or pits on the inner race, outer race, or rolling elements. Impacts from rolling elements striking these defects produce distinct fault frequencies such as BPFI, BPFO, BSF, and FTF.

Stage 4 is the complete failure, as spalls enlarge, the bearing loses load-carrying capacity. This leads to increased friction, heat generation, cage instability, and catastrophic failure of the machine.

Early fault detection includes the transition between Stages 3 and 4, which is the end goal of condition monitoring. Intervening just before complete failure prevents costly downtime and secondary damage, while minimizing material and labour costs associated with early replacement. This motivates the inclusion of a developing fault

health state in the proposed dataset, providing the critical intermediate stage between healthy and fully failed bearings that makes early detection feasible.

This process highlights why early detection is critical since catching a bearing in the crack initiation or spall formation stages can prevent costly downtime and secondary damage to shafts, gears, or housings.

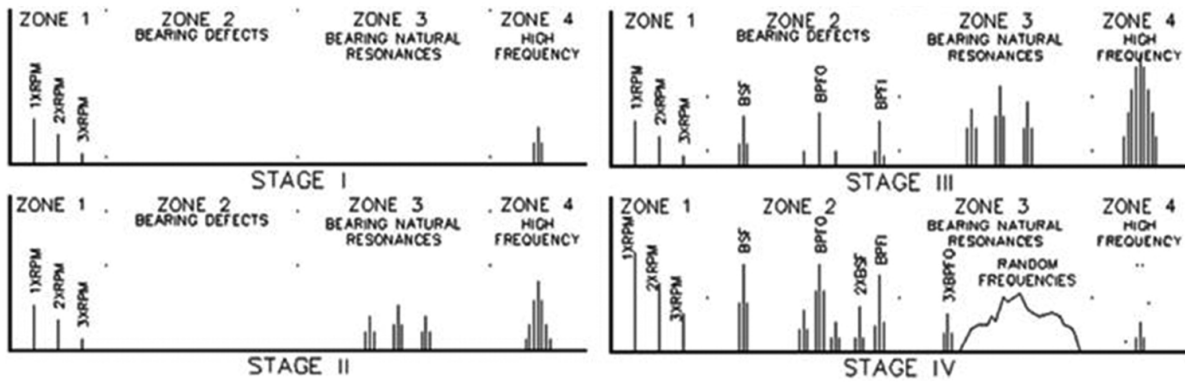


Figure 2-3. Stages of bearing failure [21]

2.1.3 Bearing Life Analysis

In addition to vibration-based fault detection, bearings are traditionally assessed using fatigue life calculations. The most common standard is the L_{10} life (equation 6), defined as the number of revolutions at which 90% of a group of identical bearings will survive under a given load without showing fatigue. It is expressed as:

$$L_{10} = \left(\frac{C}{P}\right)^p \times 10^6 \text{ revolutions} \quad (6)$$

where:

- C = basic dynamic load rating (N),
- P = equivalent dynamic bearing load (N),

- p = exponent (3 for ball bearings, 10/3 for roller bearings).

The corresponding operating life in hours can then be obtained by dividing L_{10} by the shaft rotational speed.

Although this thesis focuses on vibration and data-driven diagnosis, acknowledging L_{10} life connects the proposed work with standard bearing design and reliability analysis. Establishing reliable diagnosis is a necessary prerequisite before prognosis can be meaningfully pursued. Therefore, this thesis focuses on building that diagnostic foundation. Prognostics, which aims to predict the remaining useful life (RUL) of a machine component, requires a strong diagnostic foundation before it can be meaningfully applied. By first establishing clearer connections between fatigue life predictions and vibration-based diagnostics, this thesis ensures that the groundwork is in place for future research in prognosis.

2.2 ML for Fault Diagnosis

ML provides the ability to identify and classify machine states based on sensor data, offering an alternative to purely physics-based modelling approaches. In the context of rotating machinery, ML is primarily applied to diagnosis, which involves identifying fault types from symptoms, and prognosis, which relates to predicting how faults will evolve over time [22]. While prognosis is valuable for maintenance planning, its reliability depends on first achieving robust diagnostic accuracy.

2.2.1 Traditional versus Deep Learning

Two subbranches of ML are used in bearing research: TL and DL. TL methods focus on predefined features extracted from vibration, current, or acoustic signals.

These features (e.g., RMS, kurtosis, FFT peaks) are used as inputs for classifiers such as support vector machines (SVM), k-nearest neighbors (KNN), or random forests. These approaches are data-efficient but depend heavily on expert analyzed features, which may not generalize across different machine health states.

In contrast, DL uses multi-layer neural networks that can learn hierarchical feature representations from raw or transformed signals. Convolutional neural networks (CNNs) are effective at extracting local spatial features from spectrograms, whereas long short-term memory (LSTM), as a kind of recurrent neural network (RNN), has a capacity to learn a temporal dependency. Hybrid CNN- LSTM models integrate these advantages, which makes them effective for vibration fault diagnosis.

An important consideration in applying both TL and DL methods is how datasets are partitioned and validated. In many existing studies, data is split within-domain, meaning that training, validation, and test sets are drawn from the same source file or operating condition. While this often produces high reported accuracies, it risks providing overfit results, as the model may simply memorize noise or machine-specific signatures rather than learning generalizable fault features. A more robust approach is cross domain validation, where training and testing occur on different files, machines, or operating conditions. For example, one dataset file may be used exclusively for training, while another collected under different speeds, loads, or sensor positions is reserved for testing. This cross domain setup better reflects real-world conditions, where models must generalize to machines and environments not seen during training.

In bearing research, the use of ML and DL is framed as a classification problem. Each class corresponds to a specific machine state or component fault, such as a healthy bearing, an inner race fault, an outer race fault, a rolling element fault, or a cage fault. In some studies, classes are also extended to include fault severity levels (e.g., minor vs. severe spalls) or different operating conditions. The main purpose of the classification is to build a model that can automatically classify unseen vibration signals or frequency domain inputs into the correct class and give a reliable diagnosis of the bearing condition.

2.2.1.1 Accuracy

The classification performance for ML models is evaluated based on accuracy, which is the percentage of correctly predicted samples with respect to the total number of test samples. For example, an accuracy of 90% means that among 10 vibration segments, 9 segments are correctly classified. It directly shows the reliability of the diagnostic system to distinguish between healthy and faulty conditions. To allow for a reliable evaluation, datasets are divided into three subsets including a training set (used to fit the model), a validation set (used to monitor overfitting), and a test set (used only at the end to provide an unbiased estimate of model performance). In this thesis, the term test accuracy is used to refer to classification accuracy computed on the held-out test set, that is, the test set that is never used during training or hyperparameter selection, providing an unbiased estimate of the model's performance on unseen data. It is common in bearing fault diagnosis to observe high training accuracy, but lower validation and test accuracies.

This discrepancy indicates challenges in generalization, where the model may learn noise or test rig specific features instead of transferable fault patterns. By considering both within-domain and cross domain validation, this thesis evaluates which classification accuracy reflects true diagnostic capability in industrial-like conditions.

2.2.1.2 Hyperparameters

The input to a model, whether provided as time bins in the raw domain or frequency bins in the spectral domain, defines how bearing signals are represented for classification. A key component to shaping these inputs is the choice of hyperparameters, which are configurable settings selected before training a ML algorithm. These hyperparameters control how the model processes data. The hyperparameters used in this thesis include window size and overlap (defining how signals are segmented), sampling frequency (determining both the maximum detectable frequency and the frequency resolution after applying a FFT), batch size (number of samples processed per training step), learning rate (step size for parameter updates), and pooling type (max versus average pooling in CNN layers). These hyperparameters influence whether the diagnostic model captures sufficient information to separate healthy and faulty states. For instance, larger windows increase spectral resolution but may smooth out transient impacts, while smaller windows highlight fault impulses but risk losing frequency detail. Similarly, max pooling emphasizes distinct fault harmonics, whereas average pooling highlights global energy distributions.

In particular, this thesis provides the foundation for the definition of structured data loading approaches, presented in the next section, by defining input structures, hyperparameters and validation methods.

2.2.2 Data Loading Methods

Before signals can be passed through ML algorithms, they must first be segmented, structured, and loaded into the model in a consistent way. Traditional data loading methods often involve slicing raw time-domain vibration signals into fixed-length segments (time bins) or transforming them into the frequency domain via the FFT or spectrograms (frequency bins). These segments are then randomly shuffled and split into training, validation, and test sets. While straightforward, such random sampling risks introducing overlap between domains (e.g., training and test segments from the same file), which inflates reported accuracies without guaranteeing true generalization.

It is important to differentiate data loading methods from data augmentation methods, which are sometimes confused in the literature. Data augmentation refers to the artificial modification or enlargement of the training set by generating new samples through transformations such as adding noise, scaling amplitude, or shifting time [23]. In contrast, data loading methods do not modify the original signals, but control the segmentation, ordering, and presentation of the existing data to the model during training [24]. As both approaches work different from one another, they cannot be directly compared and should not be regarded as alternatives to each other.

As DL technologies continue to evolve, these insights highlight the importance of not only the model architecture but also the way data is presented during training. Nonetheless, traditional data handling methods continue to struggle with domain shifts caused by variability in machine behavior and operating conditions, which can degrade model performance [25].

In particular, dataset loading is often overlooked as a mechanism for improving generalization across environments, even though input structures play a vital role in determining output performance. Traditional approaches shown in Figure 2-4 are generally limited to:

- Single-channel loading, where data from one source file is segmented and sequentially presented to the network. While computationally efficient, this method often lacks variability and can result in models that are overfit to specific operating conditions.
- Multi-channel (parallel) loading, where signals from multiple sensors or sources are combined into different channels and fed to the model simultaneously. This increases diversity and robustness but at the cost of higher computational demand and increased model complexity.

It is important to emphasize that data loading is not part of the ML algorithm itself but a preprocessing step that determines how the algorithm encounters the data. In other words, loading methods control the order, source, and structure of the input presented to the model, while the ML algorithm is responsible for feature extraction and classification. This distinction is critical, because even without

changing the model architecture, different loading methods can influence generalization performance in bearing fault diagnosis.

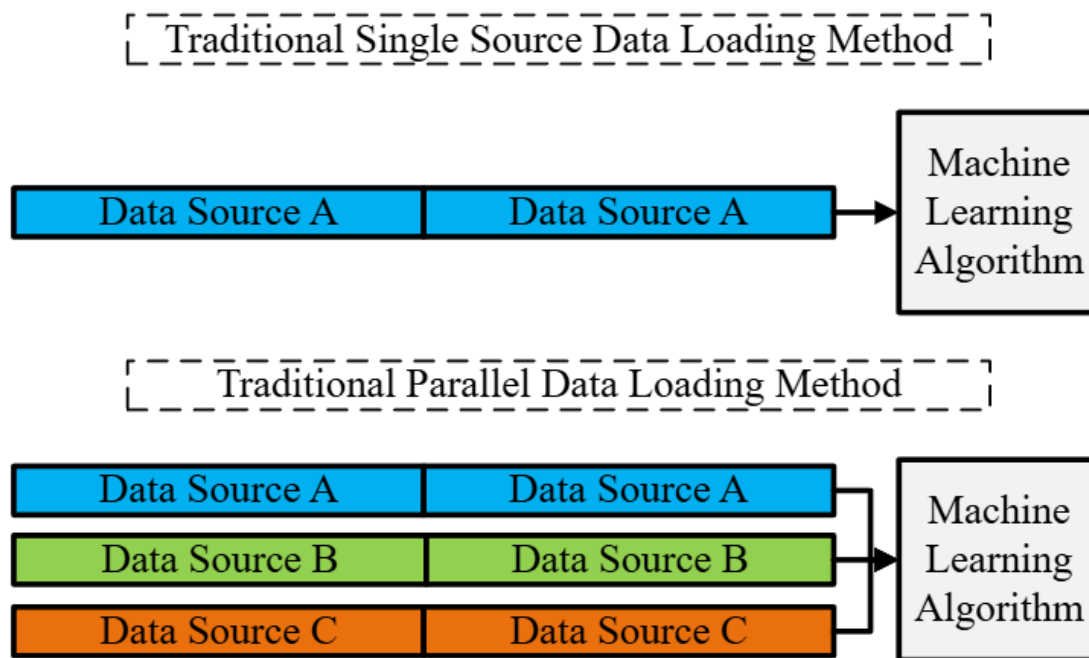


Figure 2-4. Existing data loading methods [26]

Chapter 3

Methodology

Building on the foundational concepts of roller element bearings and the importance of ML in modern fault diagnosis provided in Chapter 2, this chapter examines bearing data and current methodologies used for signal preprocessing, feature extraction, and ML model development in bearing fault diagnosis. This includes a critical look at both traditional and deep learning approaches, their hyperparameter configurations, and how they interact with bearing data. By analyzing the strengths and limitations of these existing methods, this chapter provides the context needed to explore dataset design and evaluation in the next chapter.

3.1 Datasets

This section introduces the publicly available bearing datasets used for vibration analysis in this thesis.

3.1.1 Benchmark Bearing Datasets

The Case Western Reserve University (CWRU) and triaxial bearing datasets [19], [27] capture diverse vibration signals across multiple bearing types, fault severities, and operating conditions, providing a valuable resource for evaluating model performance under varying scenarios.

3.1.1.1 CWRU Dataset

The CWRU dataset is well known for its contribution to machine fault diagnosis research. It provides a diverse set of vibration signals such as those obtained from healthy ball bearings and ball bearings with various fault conditions, including inner race, outer race, and ball faults, collected under different fault severities. This dataset has played a pivotal role in advancing the development of ML models, particularly CNNs, for fault diagnosis tasks.

However, the CWRU dataset was not specifically designed with ML applications in mind, and care must be taken to avoid misleading results. For example, issues related to data with similar load conditions may affect the generalizability of models [28], [29]. Despite this limitation, it remains a valuable benchmark for the early stages of developing a universal dataset for vibration analysis, especially when data is converted to time-frequency spectrogram images.

The CWRU data consists of sampling frequencies of 12,000 and 48,000 Hz with a sample duration of 10 seconds, and speeds around 1,720 and 1,797 RPM [19]. The test rig consists of an electrical motor seen on the left-hand side of Figure 3-1, followed by a shaft coupled to the bearing being tested and a dynamometer to control the

different loads being applied. The bearings tested with the CWRU dataset include a SKF 6205-2RS deep groove ball bearing on the drive end, and a SKF 6203-2RS deep groove ball bearing on the fan end.

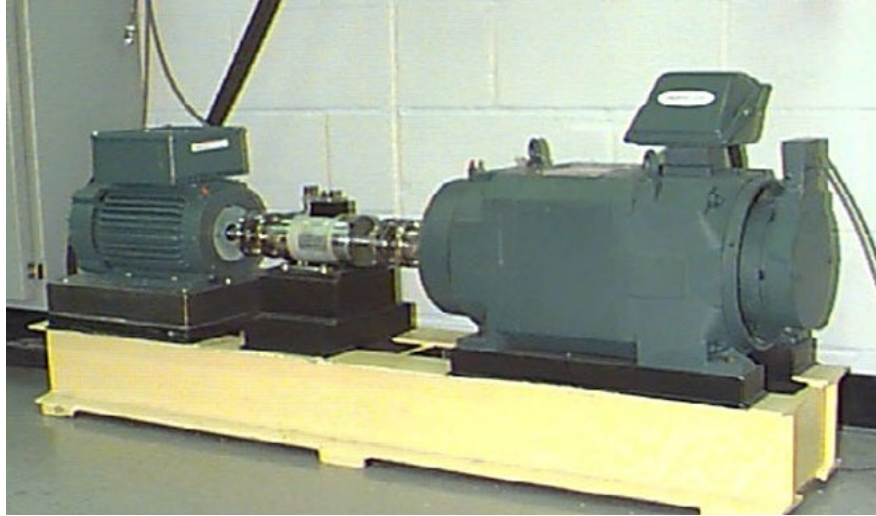


Figure 3-1. CWRU Bearing Data Test Rig [30]

3.1.1.2 Triaxial Bearing Dataset

The triaxial bearing vibration dataset enhances the diversity of available benchmark data by capturing vibration signals along three orthogonal axes (X, Y, Z) simultaneously. Data is collected from a three-phase induction motor equipped with 6204-2Z / C3 bearings under twelve different health conditions, including healthy bearings as well as inner and outer race faults at multiple severity levels (0.7 mm-1.7 mm), and operated under three distinct load conditions (100 W, 200 W, and 300 W) [27]. This design enables a detailed investigation of directional fault signatures and load-dependent vibration behavior, providing a richer feature space representation compared to single-axis datasets.

Triaxial bearing data was collected at a sampling frequency of 10,000 Hz with a sample duration of 10 seconds, and a constant speed of 1,700 RPM [27], [31]. The triaxial bearing test rig (Figure 3-2) consists of varying load conditions to give diversity.



Figure 3-2. Triaxial Bearing Data Test Rig [27]

3.1.2 Data Loading Evaluation Datasets

While Section 3.1.1 provides information on benchmark bearing datasets used to evaluate cross domain generalization in Chapter 4, this section introduces three more

datasets, spanning multiple machine types and industries, used to evaluate the proposed data loading method in Chapter 5.

3.1.2.1 Industrial Steel Slag Flow Dataset

The first dataset used for data loading evaluation is collected in industry during steel manufacturing with the purpose of detecting the flow of steel and slag (Table 3-1). This dataset is collected using a triaxial accelerometer at a sampling frequency of 6,400 Hz on the flow control shaft. This dataset represents the only dataset in this thesis collected from a real industrial environment rather than a laboratory test rig. The industrial data collection setup is shown in Figure 3-3 [32].

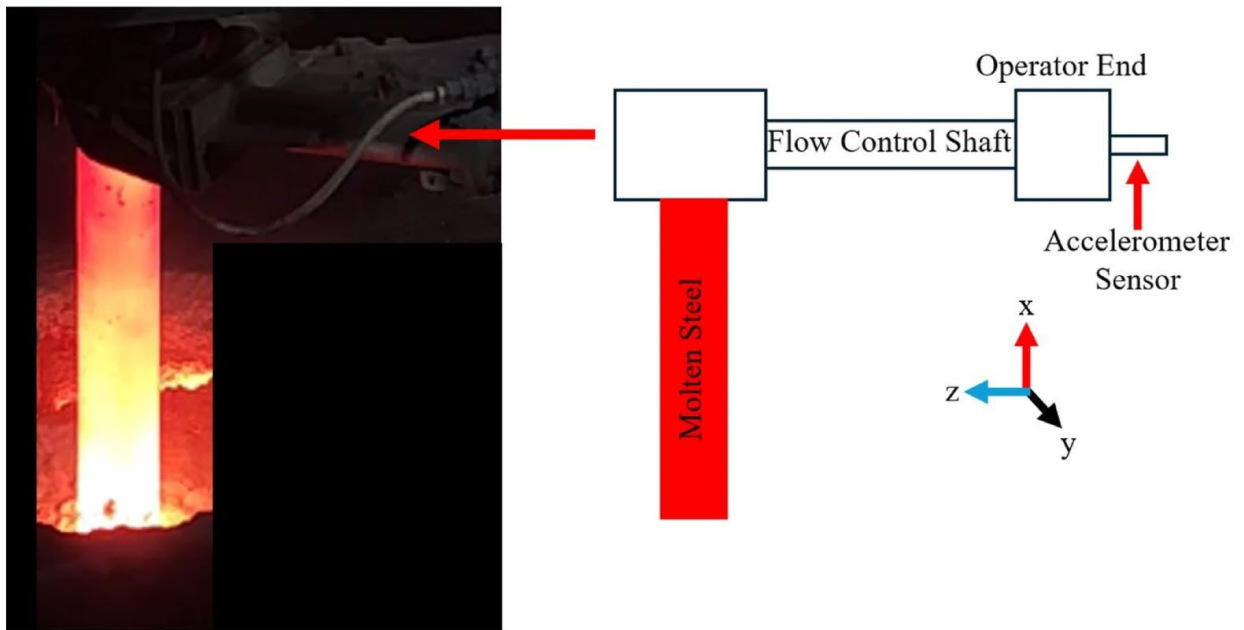


Figure 3-3. Industrial Data Collection Setup [32]

3.1.2.2 UOEMD Dataset

The UOEMD dataset contains data from an induction motor called the University of Ottawa Electric Motor Dataset (UOEMD) [33] collected using acoustic and

accelerometer sensors. Data is collected using acoustic and accelerometer sensors at 42,000 Hz from a Marathon Electric D396 motor under one healthy and seven faulty motor states including rotor unbalance, rotor misalignment, stator winding fault, voltage unbalance and single phasing, bowed rotor, broken rotor bars, and faulty bearings. The test rig setup is shown in Figure 3-4.

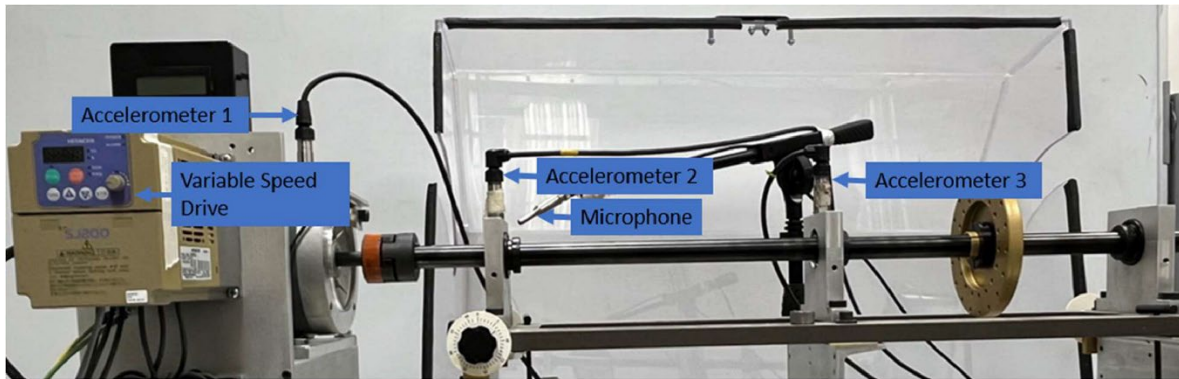


Figure 3-4. UOEMD Test Rig Setup [34]

3.1.2.3 IEEE PHM Beijing 2024 Data Challenge Dataset

The IEEE PHM Beijing 2024 Data Challenge dataset provided by Beijing Jiaotong University [35] contains data from a physically simulated train transmission system. This dataset was collected using triaxial accelerometers and a current clamp at a sampling frequency of 64,000 Hz, under operating speeds of 20 and 40 Hz and loads of 0, +10, and -10 kN. The experimental setup is depicted in Figure 3-5.



Figure 3-5. IEEE PHM Beijing 2024 Data Challenge Test Rig Setup [35]

Table 3-1 summarizes the datasets selected for data loading evaluation in Chapter 5. As can be seen, the datasets used for evaluating these data loading methods also include the CWRU and triaxial datasets described in Section 3.1.1, as well as the dataset proposed in Section 4.1 (UORED-VAFCLS).

Table 3-1 Datasets selected for this thesis

Dataset	Dataset Name	Machine Component	Specifications	Sampling Frequency	Industry	Sensor Type
Dataset 1	Steel Slag Flow Dataset [36]	Flow Control Shaft	16 different flow conditions	6,400 Hz	Manufacturing	Triaxial Accelerometer
Dataset 2	Triaxial Bearing Vibration Data [27], [31]	Bearing	SKF 6204-2Z/C3	10,000 Hz	Heavy Machinery	Triaxial Accelerometer
Dataset 3	CWRU [37]	Bearing	SKF 6205	12,000 Hz		Drive End and Fan End Accelerometers
Dataset 4	UORED-VAFCLS [38], [39]	Bearing	NSK6203ZZ, FAFNIR 203KD	42,000 Hz		Accelerometer and Microphone
Dataset 5	UOEMD [33], [34]	Induction Electrical Motor	Marathon Electric D396	42,000 Hz		Accelerometer and Microphone
Dataset 6	IEEE PHM Beijing 2024 Data Challenge [35]	Train Transmission System	20 and 40 Hz, 0,+10,-10 kN loads	64,000 Hz	Railway	Triaxial Accelerometers and Current Clamp

Across the datasets evaluated in this thesis, bearing sizes range from the SKF 6203 and SKF 6205 (CWRU dataset), SKF 6204 (triaxial dataset), and the NSK 6203ZZ and FAFNIR 203KD (UORED-VAFCLS dataset), evaluated at sampling frequencies ranging from 10,000 Hz to 48,000 Hz. This provides evidence that the proposed data loading method

generalizes across this range of bearing sizes and sampling conditions. Conclusions are not claimed to extend to bearing sizes or sampling frequencies outside those tested in this thesis.

3.2 Signal Preprocessing

Bearing data preprocessing is an important task in ML for fault classification [40], [41]. This is especially true when data available for training algorithms has not been collected under ideal conditions. Additionally, preprocessing is expected to reduce the computational time required for detecting bearing faults. Despite the availability of various methods, there is still a research gap in determining which preprocessing methods can most effectively improve the quality of ML models for sensor data, thereby ensuring the reliability and accuracy of classification models. Existing methods do not use simple statistical measures such as standardization [42] and root mean square (RMS) [43] for normalizing datasets. This section explores the use of basic statistical methods that can be used to normalize bearing data and compares them to other preprocessing methods such as the FFT.

3.2.1 Min-Max

Min-max normalization is the first preprocessing method used, as shown in equation (7) [44]. This technique preserves the original distribution of the data while ensuring all features lie within a comparable range, which is particularly useful for models that are sensitive to input scales. However, it is important to note that min-max normalization is sensitive to outliers, as extreme values can disproportionately affect the scaling. Min-max normalization scales the data to a fixed range, between - 1 and 1, as seen in equation (7).

$$x_{norm}(x) = \frac{x - x_{min}}{x_{max} - x_{min}} \quad (7)$$

where x are the individual data points that are being standardized, x_{min} and x_{max} are the minimum and maximum values within the data.

3.2.2 Z-score

The second preprocessing technique used in this thesis is normalization using standardization (Z-score normalization), as shown in equation (8) [45]. Standardization improves centrality in the dataset, where the data is preprocessed to have a mean of 0 and a standard deviation of 1 [42]. This provides a baseline for comparing the data, without which it is difficult to judge whether there is deviation in the data that could represent healthy or faulty states. Standardization ensures that different bearing conditions within a dataset can be compared more effectively. Standardization analysis uses sensor data and looks for a common value at its center, hence reducing the effect of outliers and noise. This normalization technique is

especially important for ML models, as it ensures that features used for training will lie on the same scale, boosting the model's performance.

$$x_{std}(x) = \frac{x - \mu}{\sigma} \quad (8)$$

where μ is the mean of the dataset, and σ is the dataset's standard deviation.

3.2.3 RMS

The third preprocessing method used in this thesis is RMS, as shown in equation (9) [46]. RMS is a measure that considers both the magnitude and variability of data. It is useful for detecting changes and variations in vibration signals, as well as bearing failure signatures. RMS is calculated by the square root of the mean of the points squared. A method like this enhances bigger deviations, making it easier to pick out significant changes in the signal that may indicate more substantial issues. The use of RMS in preprocessing underlines the general magnitude of signal features that are important for fault classification. RMS reduces the noise in a time-domain dataset, thereby offering more robustness against overfitting by ML models.

$$x_{RMS}(x) = \sqrt{\frac{\sum x^2}{n}} \quad (9)$$

where n is the total number of data points in the dataset.

3.2.4 Fast Fourier Transform (FFT)

The Discrete Fourier Transform (DFT) is the foundation of the FFT. It is used to convert a discrete-time signal $x[n]$ into its frequency-domain representation equivalent $X[k]$. By transforming the signal into the frequency domain, periodic fault-

related components become more visible and easier to isolate. This is essential for applying the FFT, which relies on frequency-domain data for fault characterization.

The DFT is mathematically expressed in equation (10) as:

$$X[k] = \sum_{n=0}^{N-1} x[n]e^{-i\frac{2\pi}{N}kn}, \quad k = 0, 1, \dots, N-1 \quad (10)$$

where $x[n]$ is the discrete-time signal of length N , and $X[k]$ is the resulting frequency-domain representation at the k^{th} frequency bin. This transformation breaks down the signal into its sinusoidal components, each with a specific frequency, magnitude, and phase. In this thesis, the FFT is used to compute the DFT efficiently, making it possible to analyze high-resolution datasets from both accelerometers and microphones. This step serves as the mathematical basis for FFT analysis that follows. The physical frequency corresponding to each bin k is given by $f(k) = k \times Fs/N$, where Fs is the sampling frequency in Hz and N is the window length in samples, giving a frequency resolution of $\Delta f = Fs/N$ Hz per bin and a maximum resolvable frequency of $Fs/2$ Hz.

There are various reasons why preprocessing raw bearing data using methods like RMS and the FFT is important. These methods make the effects of noise and outliers less impactful. Such normalizations and transformations are critical in ensuring that ML models are improved in their ability to learn from the data. Preprocessing also helps to identify and eliminate inconsistencies embedded within sensor readings, often greatly impacting the performance of a fault classification model. Finally, use

of simple statistical methods or the FFT for preprocessing makes the process more approachable and easier to implement, with a robust basis that can support the use of more complex methods. In general, an effective preprocessing framework for bearing data is crucial for ML fault classification in predictive maintenance and consequently reduces downtime in industry [47].

3.3 ML Architectures

3.3.1 Benchmark Bearing ML Architectures

A simple 1D CNN is first considered as a baseline architecture (shown in Table 3-2) for classifying five classes of machine conditions such as healthy, ball, inner, outer, and cage faults. CNNs are a popular architecture used for ML-based fault diagnosis. As discussed in Section 2.2.1, CNNs are effective at extracting local spatial features from vibration and acoustic signals through convolutional filtering, but they process each input window independently and do not capture how a signal evolves over time. This is a limitation for bearing fault diagnosis, where impact events from a fault repeat over time and the temporal ordering of these impacts carries diagnostic information that a CNN alone cannot represent. To address this, an LSTM layer is added following the convolutional layers, allowing the network to learn temporal dependencies across the extracted features in addition to their spatial structure [48]. Therefore, a combined CNN-LSTM architecture is proposed in Table 3-2 for classifying all classes of machine conditions using time domain accelerometer and acoustic data. Specifically, each input consists of a fixed-length, non-overlapping window of 1024 time-domain samples obtained from raw accelerometer and acoustic

signals. The window length of 1024 samples was selected based on two justifications. First, empirical testing across window lengths of 512, 1024, and 2048 samples (reported in Chapter 4) show that 1024 samples consistently provide the best balance of classification accuracy and training stability across preprocessing methods and batch sizes. Second, at the CWRU sampling rate of 48,000 Hz, 1024 samples correspond to 21.3 ms of vibration data, sufficient to observe approximately 3.5 BPFI fault cycle impacts at a dominant bearing fault frequency of approximately 162 Hz. At the lower 12,000 Hz sampling rate used in Chapter 5, the same 1024-sample window spans 85.3 ms, capturing approximately 2.6 shaft revolutions at a reduced spectral resolution. Prior to entering the network, each window is preprocessed using RMS normalization and the FFT, producing a 1D tensor of shape (1×1024) as input to the first convolutional layer. The algorithm is inspired by a previous condition monitoring algorithm for bearing detection proposed by Sehri et. al. [49]. This algorithm consists of a series of Conv1D layers with max-pooling layers that help extract hierarchical features from the input signal. The convolutional layers all have distinct filter sizes and depths and use ReLu activation functions. These features are then passed through an LSTM layer to capture temporal dependencies in the sequential data before being flattened and fed into three linear layers to produce a probability distribution over the five bearing classes. This model tries to pick out patterns from input data using convolutional operations and gets trained on categorical cross-entropy as the loss function, and is validated on different bearings of the same size with different machine states across domains. This architecture is

suitable for cross domain bearing fault diagnosis because the convolutional layers extract local fault related spectral features robust to domain shifts, while the LSTM layer captures the temporal dependencies in sequential vibration signals that are consistent across operating conditions, making the combined architecture more generalizable than either a standalone CNN or LSTM.

The selection of convolutional kernel sizes, number of layers, and feature depths in the CNN-LSTM architecture was determined through iterative empirical testing, balancing accuracy and computational efficiency for cross domain bearing fault diagnosis. Larger kernel sizes were avoided to reduce the risk of overfitting on smaller receptive fields, while varying kernel dimensions across layers was found to enhance feature diversity for fault patterns. The depth of the architecture, including the single LSTM layer for accelerometer and acoustic data, was chosen to capture sufficient temporal dependencies without incurring excessive training time or memory requirements. This configuration offered the best trade-off between performance and resource usage in preliminary experiments, making it well-suited for the industrial fault detection scenarios considered in this study.

In addition to the CNN and CNN-LSTM architectures, several existing methods from the literature are used for comparison in Chapter 4's benchmark analysis. Traditional approaches include the atrous convolutional deep inception network (ACDIN), the wide first-layer kernel deep convolutional neural network (WDCNN), AlexNet, k-nearest neighbors (KNN), and support vector machines (SVM), each of which has been applied to CWRU-based fault diagnosis using different preprocessing

methods such as the FFT and spectrograms. More recent cross domain methods, including multi-kernel maximum mean discrepancy (MK-MMD) [50], the domain adversarial neural network (DANN) [51], and transformer networks [52], are also included as advanced benchmarks. These methods are not proposed as part of this thesis but are drawn from prior work to contextualize the performance of the CNN-LSTM architecture against both traditional and state-of-the-art cross domain approaches.

To evaluate ML architectures under realistic conditions, the cross domain evaluation framework proposed by Zhao et al. [53] is adopted where training, validation, and test sets are selected from independent domains to ensure that performance results represent true generalization rather than memorization of features within each domain. This dataset splitting method ensures that overlap does not occur, avoiding data leakage. Specifically, training and validation sets are drawn entirely from individual domains (source and target), while the test set comes from an entirely separate domain that is not used during training or validation, assuring a robust evaluation.

3.3.2 Data Loading Evaluation ML Architectures

Beyond the CNN and CNN-LSTM, four additional state-of-the-art architectures are evaluated in Chapter 5 to confirm that the proposed data loading method helps to generalize across model types. The CNN-Gated Recurrent Unit (CNN-GRU) uses a gated recurrent unit instead of LSTM to achieve similar temporal modelling at a lower computational cost [54]. Residual Network 18 (ResNet18) uses residual

connections to allow for deeper feature extraction without gradient degradation [55]. Finally, the CNN-Transformer replaces recurrent layers with a self-attention mechanism to capture long-range dependencies in the input sequence [56], [57]. These architectures cover a spectrum of complexity and inductive biases such that any performance difference discussed in Chapter 5 can be ascribed to the data loading method rather than a particular model design choice.

The detailed network structures for the models used in this thesis are provided in Table 3-2.

Table 3-2 Network architectures of the methods compared

Models	Layers	Settings
CNN	Layer 1	Conv1D(1, 16, 3), BN, ReLU
	Layer 2	Conv1D(16, 32, 3), BN, ReLU, AdaptiveMaxPool1D(10)
	Layer 3	Linear(320, n_c)
CNN-LSTM	Layer 1	Conv1D(1, 32, 7), BN, ReLU
	Layer 2	Conv1D(32, 64, 5), BN, ReLU, MaxPool1D(2)
	Layer 3	Conv1D(64, 128, 3), BN, ReLU
	Layer 4	Conv1D(128, 256, 3), BN, ReLU, AdaptiveMaxPool1D(1)
	Layer 5	LSTM(256, 100, 3, dropout=0.5)
	Layer 6	Linear(200, 512), BN, ReLU, Dropout(0.5)
	Layer 7	Linear(512, 256), BN, ReLU, Dropout(0.5)
	Layer 8	Linear(256, n_c)
CNN-GRU	Layer 1	Conv1D(1, 16, 3), BN, ReLU
	Layer 2	Conv1D(16, 32, 3), BN, ReLU, AdaptiveMaxPool1D(10)
	Layer 3	GRU(32, 128, 2, bidirectional=True)
	Layer 4	Linear(256, n_c)
CNN-Transformer	Layer 1	Conv1D(1, 16, 3), BN, ReLU
	Layer 2	Conv1D(16, 32, 3), BN, ReLU, AdaptiveMaxPool1D(10)
	Layer 3	TransformerEncoder(32, 4, 128, 2)
	Layer 4	Linear(32, n_c)
ResNet18	Layer 1	Conv1D(1, 64, 7), BN, ReLU, MaxPool1D(3)
	Layer 2	BasicBlock(64, 64)×2
	Layer 3	BasicBlock(64, 128, stride=2)×2
	Layer 4	BasicBlock(128, 256, stride=2)×2
	Layer 5	BasicBlock(256, 512, stride=2)×2
	Layer 6	AdaptiveAvgPool1D(1)
	Layer 7	Linear(512, 256), ReLU, Dropout
	Layer 8	Linear(256, n_c)

3.4 Data Domain Splitting

In this work, each dataset is analyzed within itself in a cross domain manner, meaning that a subset of data (domain) is allocated exclusively for training, another

subset for validation, and the remaining for testing. This ensures that the model is evaluated on entirely unseen data (different domains), simulating industrial generalization scenarios where fault conditions differ from those in the training set. This approach should not be confused with transfer learning, where training occurs on one dataset (e.g., CWRU) and testing is performed on a different dataset (e.g., triaxial bearing data) [58]. The use of the term “cross domain” strictly refers to evaluating model generalization across distinct operating domains (i.e., machine operating states) within the same dataset, not across fundamentally different datasets [59].

Table 3-3 shows the domain splitting performed on the CWRU dataset in Section 3.1.1.1, which contains 12 domains that are organized by fault size (0.007”, 0.014”, and 0.021”), with four speed conditions per fault size captured from the drive end bearing. For the CWRU dataset, each domain consists of files for an inner race, outer race, and ball fault. This domain structure ensures that every domain used for training contains examples of all three fault classes, since a classifier can only learn to recognize a class it has seen during training. Thus, cross domain generalization is evaluated across unseen fault severities between domains, rather than across unseen fault classes. Data sample naming conventions consist of a letter representing the type of fault for the 6205 (CWRU drive end) ball bearings being tested, a second number representing a distinct bearing entity (where bearings 1-4 are healthy and the remaining numbers represent faulty bearings with different fault severities), and a third number (i.e., 0, 1, 2, 3) representing different speeds (i.e., 1797, 1722, 1750,

and 1730 RPM, respectively). An example domain consists of files such as H-1-0, I-1-1, O-6-1, B-11-1, C-16-1 in a single domain called domain 1 (Domain Name). The cross domain testing framework is conducted by combining 4 domains together and testing on another domain to see if the trained data transfers across domains for different bearings of the same size. For instance, from Table 3-3, domains 1, 2, 3, and 4 are used for training while domains 5, to 12 are used for testing one at a time following Hendriks et al.'s proposed dataset splitting for training and testing in Figure 3-6. The figure highlights two approaches for dataset splitting, including the load splitting approach, which mixes data across motor speeds, leading to data leakage while training and testing on the same bearing, whereas Hendriks et al.'s proposed splitting approach separates domains by fault size and distinct bearing entities, making it more representative of industrial situations where unseen fault sizes are encountered on distinct bearings [29]. Since the data is split by fault size rather than load, accuracies for this more difficult task were found to be very low according to Hendriks et al. [29].

Table 3-3 Domain Splitting for the CWRU Dataset 48 kHz Drive End

Domain Name	Inner Race	Outer-Race	Ball
1	I-5-0 [IR007 0]	O-8-0 [OR007@6 0]	B-11-0 [B007 0]
2	I-5-1 [IR007 1]	O-8-1 [OR007@6 1]	B-11-1 [B007 1]
3	I-5-2 [IR007 2]	O-8-2 [OR007@6 2]	B-11-2 [B007 2]
4	I-5-3 [IR007 3]	O-8-3 [OR007@6 3]	B-11-3 [B007 3]
5	I-6-0 [IR014 0]	O-9-0 [OR014@6 0]	B-12-0 [B014 0]
6	I-6-1 [IR014 1]	O-9-1 [OR014@6 1]	B-12-1 [B014 1]
7	I-6-2 [IR014 2]	O-9-2 [OR014@6 2]	B-12-2 [B014 2]
8	I-6-3 [IR014 3]	O-9-3 [OR014@6 3]	B-12-3 [B014 3]
9	I-7-0 [IR021 0]	O-10-0 [OR021@6 0]	B-13-0 [B021 0]
10	I-7-1 [IR021 1]	O-10-1 [OR021@6 1]	B-13-1 [B021 1]
11	I-7-2 [IR021 2]	O-10-2 [OR021@6 2]	B-13-2 [B021 2]
12	I-7-3 [IR021 3]	O-10-3 [OR021@6 3]	B-13-3 [B021 3]

Note: healthy data (bearings 1-4) is not included as a separate domain column as the CWRU dataset does not provide sufficient healthy bearing files to form independent

domains equivalent to the fault conditions provided herein.

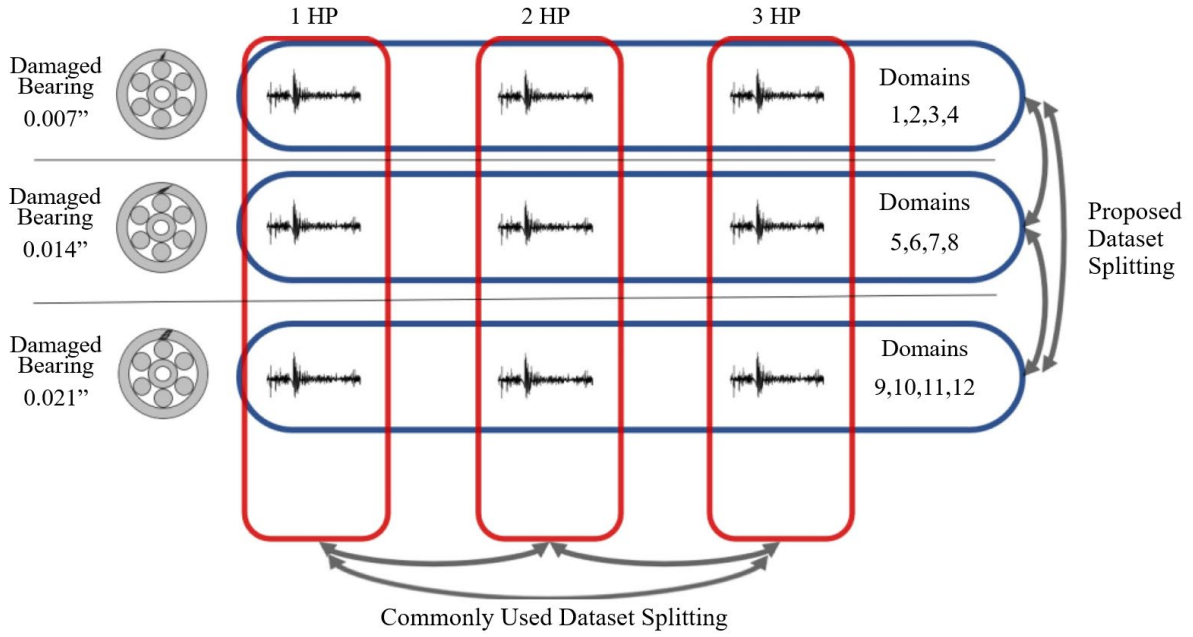


Figure 3-6. CWRU Dataset Splitting [29]

Domains for the triaxial bearing dataset in Section 3.1.1.2 are split as shown in Figure 3-7. This split allows researchers to carefully control variables, which is useful for testing specific hypotheses about model performance. Since no ball or cage faults were collected as part of this dataset, these fault types were not used for testing. The 16 working conditions (domains) are, therefore, defined only for healthy bearings, as well as inner race, and outer race fault files. Furthermore, each bearing is run under two load conditions for training (100, and 200 Watts) and one for testing (300 Watts). Since each fault severity condition is recorded from the same physical bearing across all three load levels, bearing identity is shared between the training and test domains for this dataset. This is a limitation of the triaxial bearing dataset itself rather than the domain splitting method, and this means that cross domain splits evaluate

generalization across load conditions for a known bearing, rather than generalizing to unseen bearings. A domain for this dataset consists of different bearing health states such as healthy, inner race, and outer race faults. For training, 15 domains are used and tested against the remaining domain. This is done 16 times, and the average accuracy is then taken.

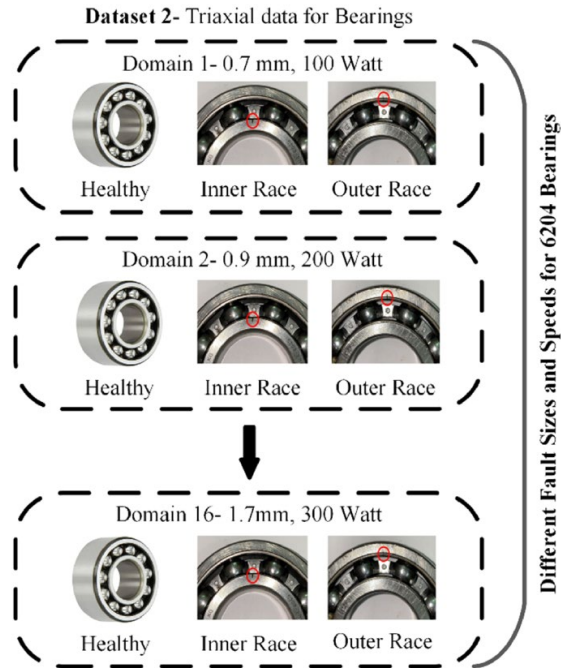


Figure 3-7. Triaxial Bearing Dataset Splitting

The datasets described in Section 3.1.2 are divided into domains as shown in Figure 3-8. These domains are chosen to reflect real-world scenarios. For Dataset 1 (steel slag flow), each of the 16 flow conditions are listed in Figure 3-8 a), and are each treated as an individual domain. This allows a model to be evaluated across the full range of flow scenarios encountered during steel manufacturing. For Dataset 5 (UOEMD, Figure 3-8 b)), domains are defined by the motor condition state, with each domain corresponding to a different operating speed. In Dataset 6 (IEEE PHM

Beijing 2024 data challenge, Figure 3-8 c)), domains are characterized by the health state of the railway transmission system component and different load conditions are employed to distinguish the domains.

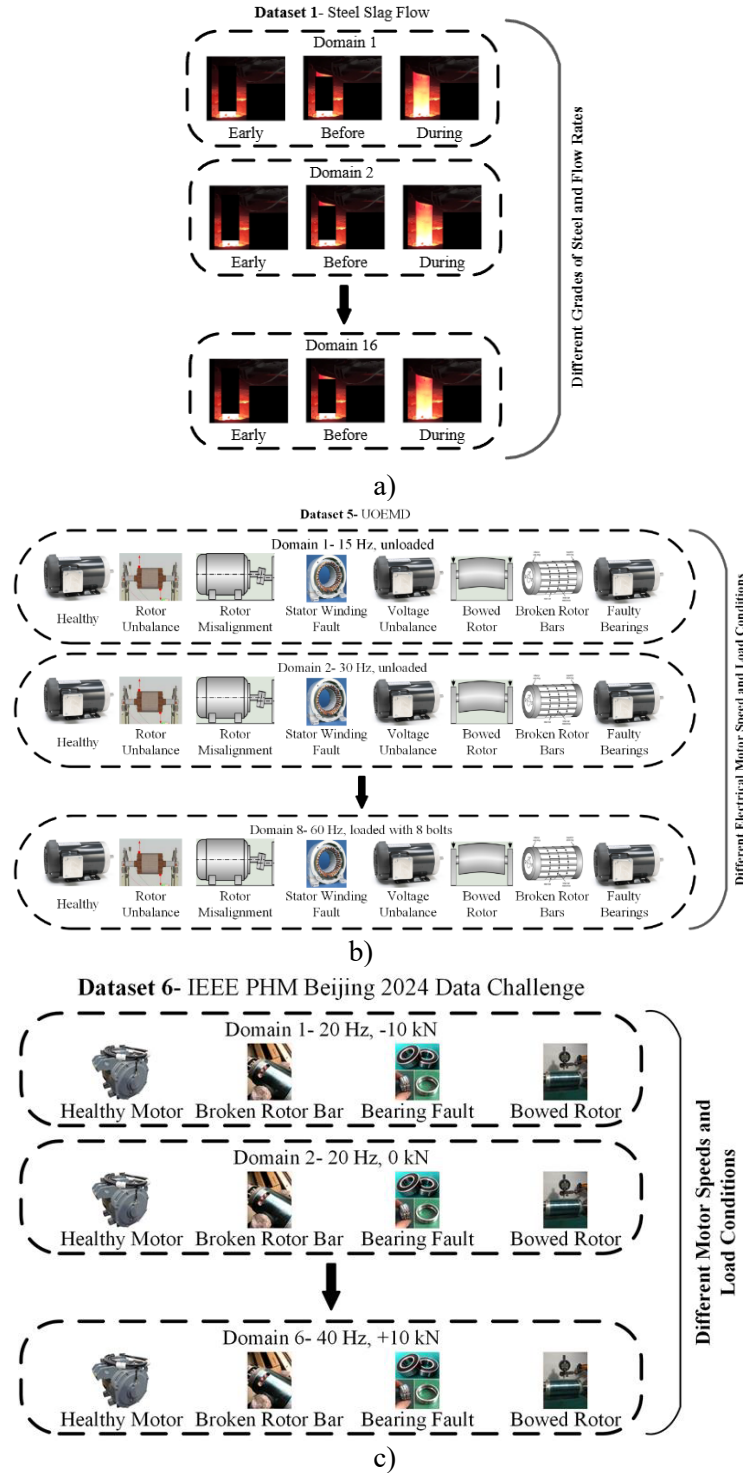


Figure 3-8. Domain loading of a) Dataset 1- Steel slag flow, b) Dataset 5- UOEMD, c) Dataset 6- IEEE PHM Beijing 2024 data challenge.

Domain splitting for Dataset 4 proposed in Section 4.1 (UORED-VAFCLS Figure 3-9) corresponds to the bearing health states (healthy, inner race, outer race, ball, and cage faults) over the 20 physically distinct bearings, as presented in Figure 3-9.

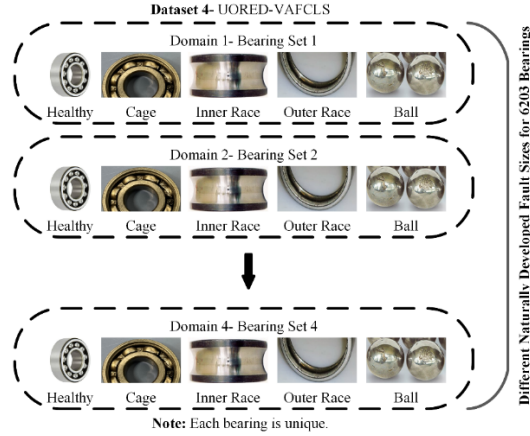


Figure 3-9. Domain loading of Dataset 4- UORED-VAFCLS.

These approaches provide a more robust evaluation of each model's performance and mitigates overfitting by ensuring that every data point is used for both training and validation, based on the assumption that training and validation datasets share the same feature distribution.

Chapter 4

Dataset Design and Analysis

To better evaluate the data loading methods described in the next chapter, this chapter proposes the development of a new bearing dataset specifically designed to address limitations identified in existing datasets and to better mimic industrial setups when applying ML fault diagnosis models. This new dataset, along with the CWRU and triaxial datasets are then evaluated across numerous preprocessing approaches and ML architectures. This work contributes to understanding and improving the breadth and quality of datasets available for evaluating ML-based fault diagnosis models and methods.

4.1 Proposed Dataset Design

The University of Ottawa Rolling-element Dataset – Vibration and Acoustic Faults under Constant Load and Speed conditions (UORED-VAFCLS) dataset is designed to have several advantages over existing public datasets that will make it valuable for ML fault diagnosis. Unlike seeded fault datasets such as CWRU and the triaxial bearing dataset, the faults in UORED-VAFCLS are developed naturally through each bearing’s own failure progression by design, producing realistic vibration and acoustic signatures that better reflect industrial conditions. Moreover,

the simultaneous collection of acoustic data from the same bearing under controlled operating conditions, above and beyond the accelerometer data that is typically collected, provides a second synchronized signal source. This makes it possible to load multi source data from two distinct sources, which is not possible with single sensor datasets. Three health states are also collected, including healthy, developing fault, and faulty, which directly support early fault detection, a primary goal of condition monitoring as indicated in Chapter 2. Furthermore, bearing-to-bearing variability is provided by including 20 distinct bearings. This ensures that cross domain evaluation is meaningful, with each bearing representing an independent domain rather than a repeated measurement of the same physical entity. These explicit design considerations make UORED-VAFCLS unique and highly suitable for validating data loading methods introduced in Chapter 5, where the availability of two synchronized and distinct sensors helps demonstrate the benefits of alternating between diverse data sources during training.

A nominal constant speed of 1,750 RPM is used throughout this dataset with an applied load of 400 N for healthy, inner race, outer race and cage faults and 0 N for ball faults to accelerate natural fault development. The controlled operating conditions remove variability due to changing speed or load, isolating the effect of fault progression. This means that changes in the fault signals can be attributed to bearing degradation rather than to changes in operating conditions. This gives a clean baseline for evaluating data loading methods.

4.1.1 Experimental Setup

Data collection was performed using the UORED-VAFCLS test rig. The experimental setup is shown in Figure 4-1. The experimental test rig consists of a single-phase motor mounted on a rigid plate supported by anti-vibration mounts. The motor shaft is stepped up using a shaft adapter, on which an SKF E22206 spherical roller bearing is mounted to withstand the load applied by the cantilever beam. The load applied by the beam is controlled via a lead screw. The motor is driven at a constant nominal speed of 1,750 RPM. The motor shaft is supported by two internal NSK 6203ZZ steel ball bearings. The drive-end bearing model was changed after 5 tests; the first 5 tests included the 6203ZZ bearings, and the subsequent 15 tests consisted of FAFNIR 203KD bearings. The drive end bearing seals were removed, and the bearings were degreased to accelerate deterioration during testing, allowing faults to develop naturally through the bearing's own failure progression rather than through artificial seeding. While removing a bearing's seals and grease accelerates the onset of failure relative to a bearing operating under normal lubricated conditions, the resulting faults still develop through the bearing's own progressive wear mechanisms rather than through artificially seeded or machined defects. Therefore, in this thesis, the term "naturally developed" refers to the mechanism of fault formation, not a normal unaccelerated timeline. Bearings were then replaced after each test. The drive end bearing is the bearing for which data is collected. The bearing specifications used in the collected datasets are provided in Table 4-1.

Table 4-1 Bearing Specifications

Bearing Model	Pitch Diameter	Ball Diameter	Number of Balls
NSK 6203ZZ	28.50 mm	6.77 mm	8
FAFNIR 203KD	28.50 mm	6.77 mm	8

The test rig shown in Figure 4-1 has six sensors as follows: (1) accelerometer, (2) microphone, (3) load cell, (4) hall effect sensor, (5) motor temperature thermocouple, (6) room temperature thermocouple. The accelerometer (PCB, model 623C01) is mounted directly on the drive end bearing using a magnet by altering the motor casing, as shown in Figure 4-2 b). The microphone (PCB, model 130F20) is placed within 2 cm of the motor drive-end bearing being tested but is supported independently of the motor. The load cell (OMEGA, model LCM302) is mounted between the cantilever arm and the SKF E22206 bearing used to transfer the load to the motor shaft. The hall effect sensor (OMEGA, model OMDC-MPU-A), used to determine the motor's rotational speed, is mounted on an adjustable plastic support and placed 2 mm away from a two-toothed gear. The two-toothed gear is mounted to the end of the drive shaft adapter, as seen in Figure 4-2 a). The thermocouples (OMEGA, model KTSS-HH) are used to determine the temperature differential of the bearing by collecting the room temperature and the bearing outer race temperature.

The experimental setup allows faults to be created in the motor drive end bearing and for the collection of data with a minimal signal-to-noise ratio. The accelerometer sensor placement was tested on both the outside and the inside of the motor casing (directly mounted on the drive end bearing being tested itself). The accelerometer mounted inside the motor casing was found to have significant noise reduction when compared to outside the casing without suffering from electromagnetic field

problems. Therefore, all provided data is recorded when the accelerometer is mounted inside the motor casing. This allows researchers to focus on analyzing a clean bearing signal with reduced noise.

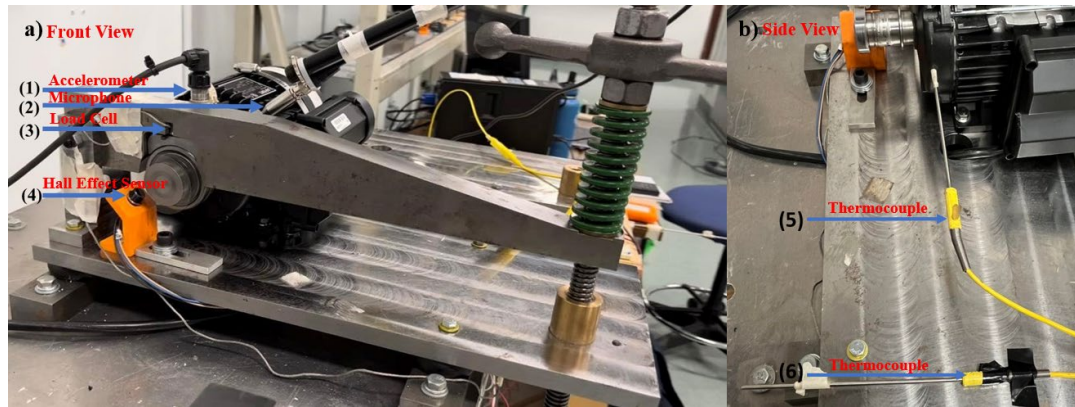


Figure 4-1. UORED-VAFCLS Test Rig Set Up in Operation a) Front View b) Side View

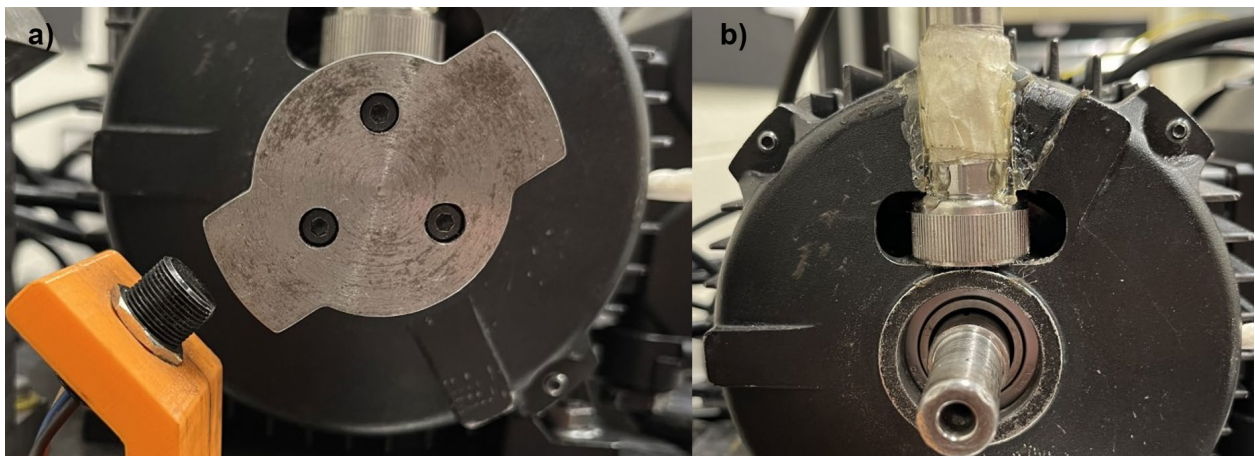


Figure 4-2. a) Two-Toothed Gear, b) Accelerometer Placement (Front View Close-up) [39]

4.1.2 Data Acquisition System

A National Instruments USB-6212 data acquisition system (DAQ) is used to connect sensors to the computer. A single DAQ unit is used to acquire all sensor channels simultaneously. This is a limitation based on the available hardware rather than an explicit design choice. While this ensures synchronization across all sensors, it also constrains the sampling rates for each sensor when multiple channels are acquired concurrently since the available bandwidth of the DAQ is shared across all channels and must be the same for each channel. The accelerometer and the microphone sensors are attached to a PCB Piezotronics 482C signal conditioner. The accelerometer is used to collect vibration signals, the microphone collects acoustic signals, the load cell collects load data, the hall effect sensor is used to collect the rotational speed of the motor shaft, and the thermocouple sensors are used to collect room and bearing temperatures. Temperature data is collected and included in the raw dataset files to provide complementary context alongside the primary vibration and acoustic signals. However, it is not used as an input feature in the classification experiments presented in Chapters 4 and 5.

The raw hall effect sensor data is converted to RPM by converting the voltage to frequency = $1 / (2 \times 60)$ (two gear teeth per rotation). The load cell, accelerometer, and thermocouple values are captured in V , the data is then converted to N , $\frac{m}{s^2}$, and $(^{\circ}C)$, respectively, using the sensitivity conversion rates provided by each manufacturer. The temperature differential is calculated by subtracting the room temperature from the bearing temperature. Therefore, the data presented in each raw data file have

the following units: vibration ($\frac{m}{s^2}$), acoustic sound (V), load (N), rotational speed (RPM), and temperature ($^{\circ}C$).

4.1.3 Data Collection

The proposed dataset’s design is rooted in the bearing mechanics established in Chapter 2. Specifically, the sampling rate, data collection duration, and fault categorization are each justified by bearing signature theory, consisting of fault frequency characteristics, failure progression stages, and harmonic requirements for reliable fault identification. To mimic typical industrial setups for bearing fault detection, the PCB 623C01 accelerometer and the PCB 130F20 microphone are selected with a maximum frequency limited to 15,000 Hz and 20,000 Hz, respectively. The accelerometer’s upper bound is selected to capture as much high-frequency resonant content as the available instrumentation allows, consistent with the diagnostic importance of the bands discussed in Section 2.1.1. This bandwidth provides substantial margin above the characteristic fault frequencies discussed in Section 2.1.1 (up to approximately 144 Hz for BPFI at 1,750 RPM) while also capturing structural resonances in the kHz-range excited by fault impacts, which carry diagnostic information beyond the fundamental fault frequencies alone. Meanwhile the microphone is selected to include the audible range. A sampling rate of 42,000 Hz is chosen based on the data acquisition system and the Nyquist theorem [60]. Based on the requirements of many DL algorithms (i.e., the minimum limiting factor in this dataset design), a high number of samples is required. Moreover, using the defined sampling frequency, 10 seconds of data collection is deemed suitable

based on two criteria. First, at 1,750 RPM the four characteristic fault frequencies for the 6203ZZ and 203KD bearings used in the test rig setup are FTF $\approx$ 11.1 Hz, BSF $\approx$ 57.9 Hz, BPFO $\approx$ 89.0 Hz, and BPFI $\approx$ 144.4 Hz. FTF is the lowest of the four characteristic frequencies, and therefore, the limiting case for cycle coverage. A 10 second window captures approximately 111 FTF cycles, well beyond the 10-harmonic minimum established in Section 2.1.1. The remaining fault frequencies are captured with substantially greater margin, at approximately 579, 890, and 1,444 cycles for BSF, BPFO, and BPFI, respectively. Second, at 42,000 Hz this duration yields 420,000 samples per file, providing approximately 410 non-overlapping 1024-sample windows per file, sufficient to generate multiple independent training samples per health state while remaining practical for repeated data collection across 20 bearings.

Each bearing has natural and stochastic inclusions that occur during material processing and component manufacturing that locally increase material stresses and lead to subsurface cracks [61]. Therefore, each set of data (i.e., 420,000 samples) provides a different result. Nonetheless, the data collection procedure can be replicated due to constant variables selected with respect to the sampling rate (42,000 Hz) and the number of samples collected per sample (420,000). Moreover, the experimental test setup is designed to ensure that bearing failures occur in a similar way each time (irrespective of the failure mode) and that data collection is consistent across all experiments. Finally, the dataset is created with 5 distinct bearings per fault type (inner race, outer race, ball and cage) for a total of 20 rolling element bearings, providing enough bearing-to-bearing variation for cross domain evaluation

where each bearing instance is part of an individual domain. Each bearing's data is collected at three stages of failure progression, including healthy, developing fault, and faulty, as established in Chapter 2, allowing the dataset to support early fault detection in addition to bearing-independent cross domain evaluation. This directly addresses the bearing-independence and health-state limitations identified in Section 3.1, in which the CWRU dataset relies on seeded rather than naturally developed faults, and the triaxial dataset reuses the same physical bearing across load domains. Instead, this dataset uses naturally developed faults across 20 physically distinct bearings, ensuring that training and test domains are never drawn from the same physical bearing instance.

Each set of bearing data is selected from a pool of 50 files collected over time, and each set of data is plotted with an FFT to ensure the samples have a clean signal that allow for fault-type validation. The faults are both identified visually and validated with the plotted FFTs. Three bearing conditions (healthy, developing fault, and faulty) are selected for data collection. Healthy data (selected within the first 5 files) is collected as a baseline for comparison with the developing fault data (selected within files 25 to 30) to see if a diagnosis can be made at the bearing's half-life point. Faulty data (selected within files 45 to 50) is also collected for bearing fault diagnosis.

4.1.4 Raw Dataset Design

The data for healthy bearings, as well as bearings with inner, outer, and cage faults, are recorded under a consistent nominal load of 400 N. For ball fault data, no load is applied since degreasing the bearings removes lubrication and causes the

rolling elements to rub directly against the raceway even without an applied load. This allows ball faults to develop naturally within a reasonable test timeframe without requiring an additional load to accelerate degradation.

Each data sample measures 10 seconds in length and is collected at a sampling rate of 42,000 Hz. In the resulting data file, the first column provides accelerometer data, the second column is acoustic data, the third column is motor speed, the fourth column is the load, and the fifth column is temperature data. Raw data is provided as time series vibration and acoustic amplitudes, whereas motor speeds, applied load, and temperature are provided as is. Each raw data sample is labeled with the format {Letter}-{Number}-{Number}.

The letter in the dataset numbering system indicates the final condition of the bearing being tested. Specifically, "H" denotes a healthy bearing, "I" indicates an inner race fault, "O" represents an outer race fault, "B" signifies a ball fault, and "C" indicates a cage fault.

The first number in the dataset numbering identifies the specific bearing that is tested. The second number indicates the condition of the bearing's health, where "0" indicates a healthy bearing, "1" signifies a developing fault, and "2" denotes a faulty bearing.

For example, the data sample labeled "I-2-1" corresponds to an inner race fault in bearing 2 with a developing fault, whereas "I-2-2" represents an inner race fault in bearing 2 that is faulty. Similarly, "O-6-1" stands for an outer race fault in bearing 6

with a developing fault, while "O-7-2" indicates an outer race fault in bearing 7 that is faulty.

Table 4-2 provides a detailed breakdown of the dataset numbering scheme.

Table 4-2 Dataset Numbering

Bearing health condition	Constant load and speed 0 N, 400 N, and 1,750 RPM				
Healthy	H-1-0	H-2-0	H-3-0	H-4-0	H-5-0
	H-6-0	H-7-0	H-8-0	H-9-0	H-10-0
	H-11-0	H-12-0	H-13-0	H-14-0	H-15-0
	H-16-0	H-17-0	H-18-0	H-19-0	H-20-0
Developing fault (inner race)	I-1-1	I-2-1	I-3-1	I-4-1	I-5-1
Faulty (inner race)	I-1-2	I-2-2	I-3-2	I-4-2	I-5-2
Developing fault (outer race)	O-6-1	O-7-1	O-8-1	O-9-1	O-10-1
Faulty (outer race)	O-6-2	O-7-2	O-8-2	O-9-2	O-10-2
Developing fault (ball)	B-11-1	B-12-1	B-13-1	B-14-1	B-15-1
Faulty (ball)	B-11-2	B-12-2	B-13-2	B-14-2	B-15-2
Developing fault (cage)	C-16-1	C-17-1	C-18-1	C-19-1	C-20-1
Faulty (cage)	C-16-2	C-17-2	C-18-2	C-19-2	C-20-2

Figure 4-3 provides the block diagram for the dataset design, including each bearing state and manufacturer.

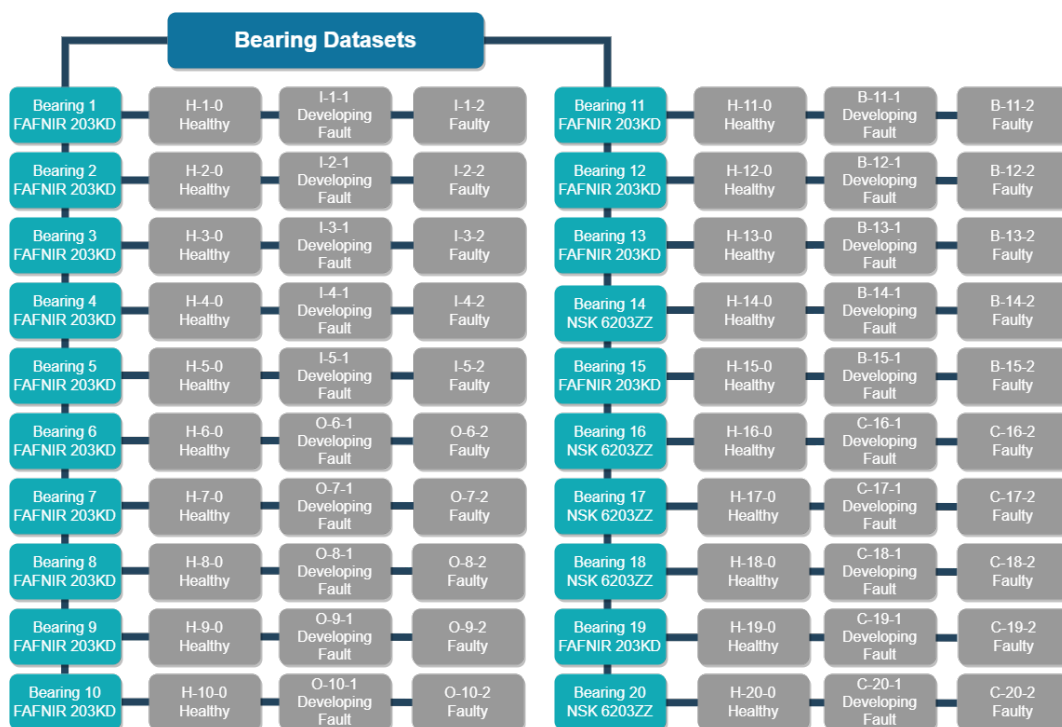


Figure 4-3. Dataset Design Block Diagram

A sample of O-7-2 (bearing 7 outer race fault) and H-7-0 (bearing 7 healthy) is shown in Figure 4-4 a) and b) (vibration data), Figure 4-5 (acoustic data). Figure 4-6 a) and b) provides the FFT for the same accelerometer samples, showing a potential outer race fault when compared to a healthy bearing.

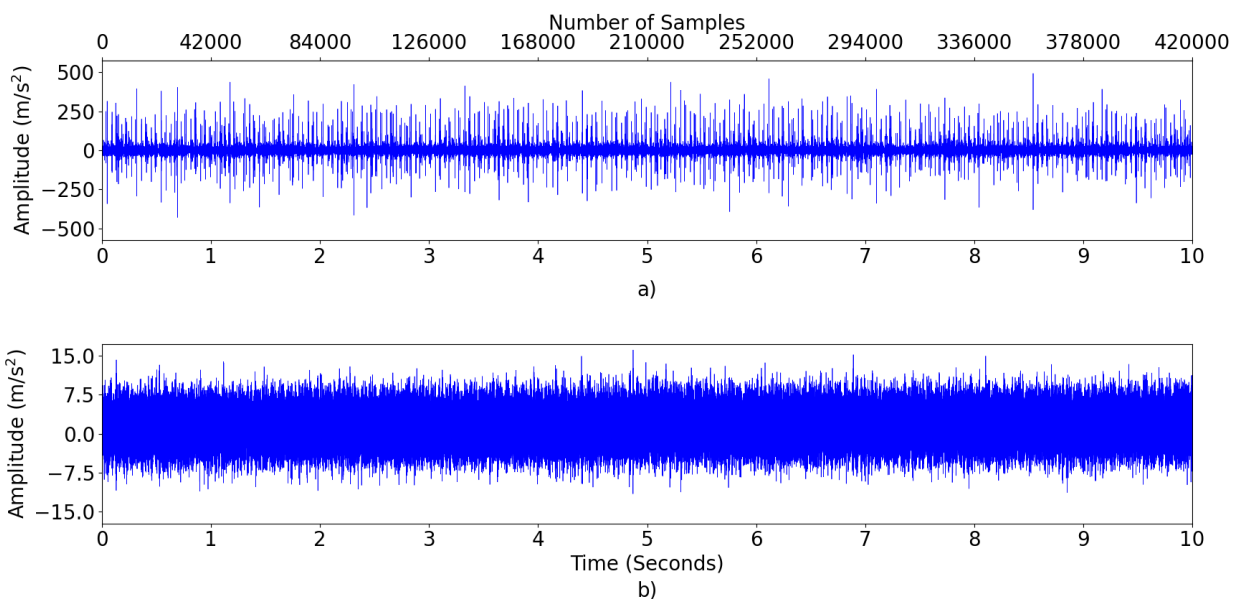


Figure 4-4. Accelerometer Data for a) O-7-2 and b) H-7-0

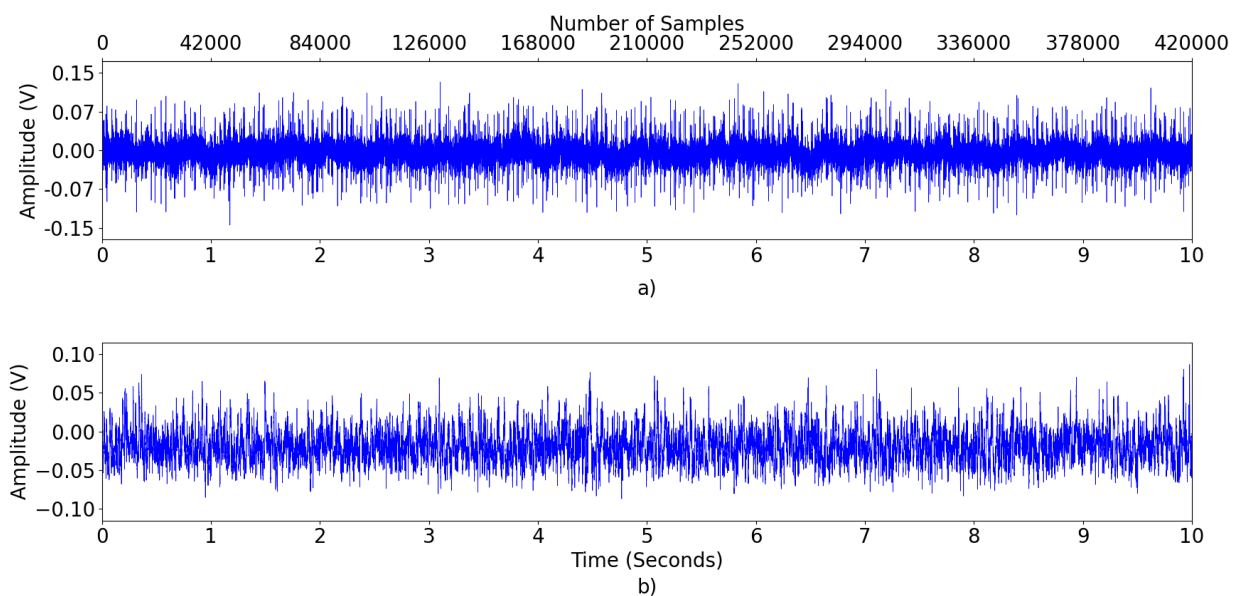
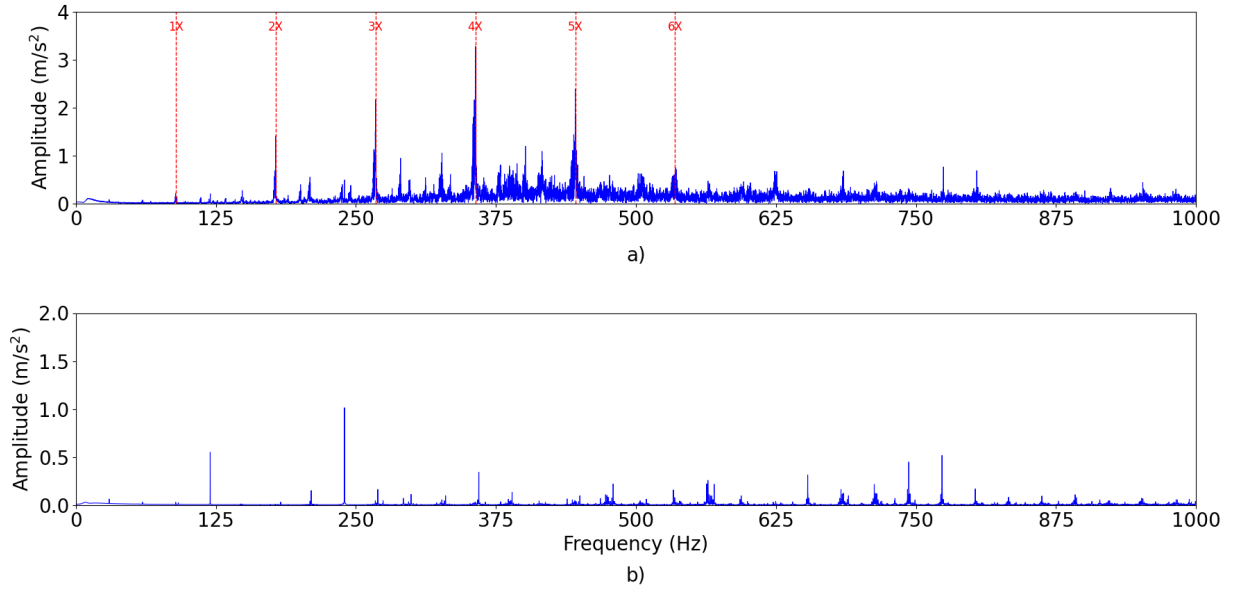


Figure 4-5. Acoustic Data for a) O-7-2 and b) H-7-0**Figure 4-6.** FFT Validation for a) O-7-2 and b) H-7-0

4.2 Bearing Dataset Analysis

This section conducts preliminary experiments on existing datasets, as well as the newly proposed dataset, to gain insight into how they perform with common preprocessing and ML architectures.

4.2.1 CWRU Analysis

Drive end accelerometer signals at a sampling rate of 48,000 Hz from the CWRU dataset are used for all experimental input configurations. Raw time-domain signals are segmented into fixed-length, non-overlapping windows prior to preprocessing, and each window is treated as one independent training sample. The input signal column in Table 4-3 refers to the number of time-domain samples per window, where

512 samples correspond to 10.7 ms, 1024 samples to 21.3 ms, and 2048 samples to 42.7 ms at the given sampling rate. These three window lengths were selected to investigate the trade-off between temporal context and training efficiency. Shorter windows produce more training samples per file but capture fewer fault cycles, while longer windows preserve more of the temporal structure at the cost of a reduced sample count. Selected preprocessing methods (min-max, Z-score, RMS, or FFT) are applied independently to each window before it is passed to the network as a 1D tensor of shape $(1 \times N)$, where N is the window length.

Table 4-3 demonstrates the performance of different ML architectures and parameters when used with the CWRU dataset’s accelerometer data. Domains 1 to 8 are used for training and validation while domain 9 is used for testing. The proposed 1D CNN-LSTM model exhibits fair stability with test accuracies exceeding 60% in most cases and a low standard deviation, particularly when using RMS and FFT preprocessing. Optimal performance was achieved with a batch size of 128 and an input signal length of 1024, reaching a test accuracy of $67.5\% \pm 2.2$ using FFT preprocessing, surpassing the best RMS result of $64.0\% \pm 2.4$. These findings suggest that the model configuration with FFT preprocessing, a batch size of 128, and an input signal length of 1024 should be utilized for further accelerometer data testing, as it offers the best combination of accuracy and stability. These results demonstrate an improvement over traditional methods when used with the more difficult data splitting framework proposed by Hendriks et al. [29], supporting the use of the combined CNN-LSTM architecture. This highlights the potential benefits of

integrating CNNs with RNNs and preprocessing for improved fault detection performance with the CWRU dataset.

Table 4-3 Accelerometer Results of Different ML Models for the CWRU Dataset

Model Type	Preprocessing	Test Accuracy (%)	Batch Size	Input Signal
1D CNN	Min-max	43.1±2.5	64	512
		42.8±3.4	64	1024
		45.4±3.9	64	2048
		42.9±3.0	128	512
		43.5±2.3	128	1024
		46.7±2.9	128	2048
	Z-score	44.3±3.5	64	512
		44.8±2.3	64	1024
		49.0±3.9	64	2048
		45.7±3.9	128	512
		44.6±3.4	128	1024
		47.4±3.4	128	2048
	RMS	43.8±1.6	64	512
		44.3±3.3	64	1024
		41.4±3.8	64	2048
		45.8±2.9	128	512
		48.0±2.3	128	1024
		47.8±3.9	128	2048
	FFT	46.1±1.7	64	512
		47.3±2.9	64	1024
		43.7±3.3	64	2048
		47.9±3.0	128	512
		50.3±3.0	128	1024
		50.1±3.5	128	2048
1D CNN-LSTM	Min-max	54.2±1.9	64	512
		55.2±2.3	64	1024
		56.9±3.7	64	2048
		54.3±2.6	128	512
		57.1±2.8	128	1024
		55.7±2.1	128	2048
	Z-score	56.1±1.7	64	512
		57.3±1.7	64	1024
		58.1±4.5	64	2048
		56.0±1.5	128	512
		58.0±3.0	128	1024
		58.7±3.8	128	2048
	RMS	55.7±1.3	64	512
		62.6±2.3	64	1024
		63.6±4.2	64	2048
		61.0±2.7	128	512
		64.0±2.4	128	1024
		63.2±3.6	128	2048
	FFT	57.1±1.9	64	512
		63.5±2.6	64	1024
		64.8±3.5	64	2048
		63.4±2.8	128	512
		67.5±2.2	128	1024
		65.6±2.4	128	2048

Note: Training and validation accuracy exceeded 95% across all model configurations, indicating sufficient model capacity. Therefore, test accuracy is the primary metric for cross domain generalization.

Results in Table 4-4 highlight the challenges of cross domain fault diagnosis, particularly when handling data with different fault severities. The table compares existing results of traditional cross domain methods, such as the atrous convolutional deep inception network (ACDIN), the wide first-layer kernel deep convolutional neural networks (WDCNN), AlexNet, KNN, and SVM, all of which use raw data or implement different preprocessing methods like the FFT or spectrogram. Additionally, advanced cross domain methods, such as multi-kernel maximum mean discrepancy (MK-MMD) [50], the domain adversarial network (DANN) [51], and transformer networks [52] are tested for comparison. Despite the variety of approaches, these models consistently achieve test accuracies below 60% for the more difficult task of domain splitting across fault severities, as shown in the literature [28], [29]. This more difficult data splitting task, based on different fault sizes across domains, is closer to what would be observed in industry. In contrast, the CNN-LSTM model achieves an average test accuracy of $64.0\% \pm 2.4$ with RMS preprocessing and $67.5\% \pm 2.2$ with FFT preprocessing, demonstrating its relative improvement over both traditional and recent advanced methods including MK-MMD, DANN, and transformer networks, which all remain below 58%.

The input representations used by each method differ in their implementation. The CNN-LSTM model proposed in this thesis uses 1D time-domain or frequency-

domain accelerometer inputs of 1024 samples (21.3 ms at 48,000 Hz) with RMS and FFT preprocessing. Table 4-4 extends this comparison by evaluating the CNN-LSTM against existing cross domain methods from the literature. Traditional methods marked with *(ACDIN, WDCNN, AlexNet, ResNet) are reproduced from Hendriks et al. [29], where ACDIN and WDCNN operate on 1D raw or FFT-transformed signals of the lengths shown, while AlexNet and ResNet operate on 2D spectrogram image inputs derived from 11,500 sample windows, where the input signal value for these methods reflects the original time-domain sample count used to generate the spectrogram, not a 1D tensor length. Methods marked with **(KNN, SVM, RF, 1D CNN) are reproduced from Rauber et al. [28] and use raw time-domain signals with handcrafted features. In this case, no fixed window length was reported. Advanced methods (MK-MMD, DANN, Transformer) were re-implemented in this work using the same 1024 sample 1D input as the CNN-LSTM to ensure a fair comparison of cross domain learning methods under identical input conditions. Computational time is reported for the methods implemented in this work only.

Table 4-4 Accelerometer Results for Existing Fault Severity Domains of the CWRU Dataset

Cross Domain Methods	Model Type	Preprocessing	Domain Train Validation	Domain Tested	Test Accuracy (%)	Batch Size	Input Signal	Time (s)
Traditional	ACDIN*	Raw	D, E, F	D, E, F	37.1 ± 12.7	N/A	5118	N/A
	ACDIN*	FFT			40.5 ± 11.0	N/A	10,236	N/A
	WDCNN*	Raw			36.2 ± 9.4	N/A	2048	N/A
	WDCNN*	FFT			50.0 ± 6.3	N/A	4096	N/A
	AlexNet*	Spectrogram			47.3 ± 11.0	N/A	11,500	N/A
	ResNet*	Spectrogram			53.1 ± 6.5	N/A	11,500	N/A
	KNN**	Raw	Average of 0.007, 0.014, 0.021	0.007, 0.014, 0.021	50.7 ± 1.2	N/A	N/A	N/A
	SVM**	Raw			49.7 ± 2.5	N/A	N/A	N/A
	RF**	Raw			52.5 ± 1.1	N/A	N/A	N/A
	1D CNN**	Raw			53.2 ± 1.9	N/A	N/A	N/A
	CNN-LSTM	RMS			64.0 ± 2.4	128	1024	55
	CNN-LSTM	FFT			67.5 ± 2.2	128	1024	140
Advanced	MK-MMD	RAW			57.3 ± 2.8	128	1024	80
	DANN	RAW			56.6 ± 3.7	128	1024	75
	Transformer	RAW			57.0 ± 3.3	128	1024	90

*[29], **[28]

Unlike existing approaches, the proposed model integrates preprocessing using RMS and the FFT, which contributes to stabilizing performance across domains. This preprocessing step enhances the model's ability to generalize features, reducing the impact of domain-specific noise and inconsistencies. For example, while network models in prior studies achieve average accuracies below 55% on the same dataset across fault severities, the proposed model exceeds this, achieving accuracies above 60% in most domains. This improvement underscores the benefits of integrating spatial and temporal feature extraction with preprocessing.

Figure 4-7 visually represents the comparative performance of these models, as summarized in Table 4-4. The bar chart demonstrates that traditional approaches such as ACDIN, WDCNN, and AlexNet using spectrograms as an input alone exhibit a wide range of average test accuracies, with standard deviation bars indicating the variability in their performance, while advanced methods perform slightly better

than most traditional methods in terms of test accuracy. Additionally, when computational times in Table 4-4 for the CNN-LSTM method are compared to the advanced methods, the CNN-LSTM combined with RMS preprocessing outperforms all three. Unfortunately, the CNN-LSTM with FFT preprocessing does take much longer to compute. The figure highlights that while certain methods like ResNet with spectrogram inputs reach a test accuracy of $53.1\% \pm 6.5$, none surpass 55% accuracy consistently across domains. On the other hand, the proposed CNN-LSTM model, clearly stands out with a higher accuracy of $64.0\% \pm 2.4$ with RMS preprocessing and $67.5\% \pm 2.2$ with FFT preprocessing, both of which surpass all traditional and advanced methods shown. The figure shows the improvement in test accuracy by the proposed model and aligns with the numerical findings, visually reinforcing the advantages of the CNN-LSTM method.

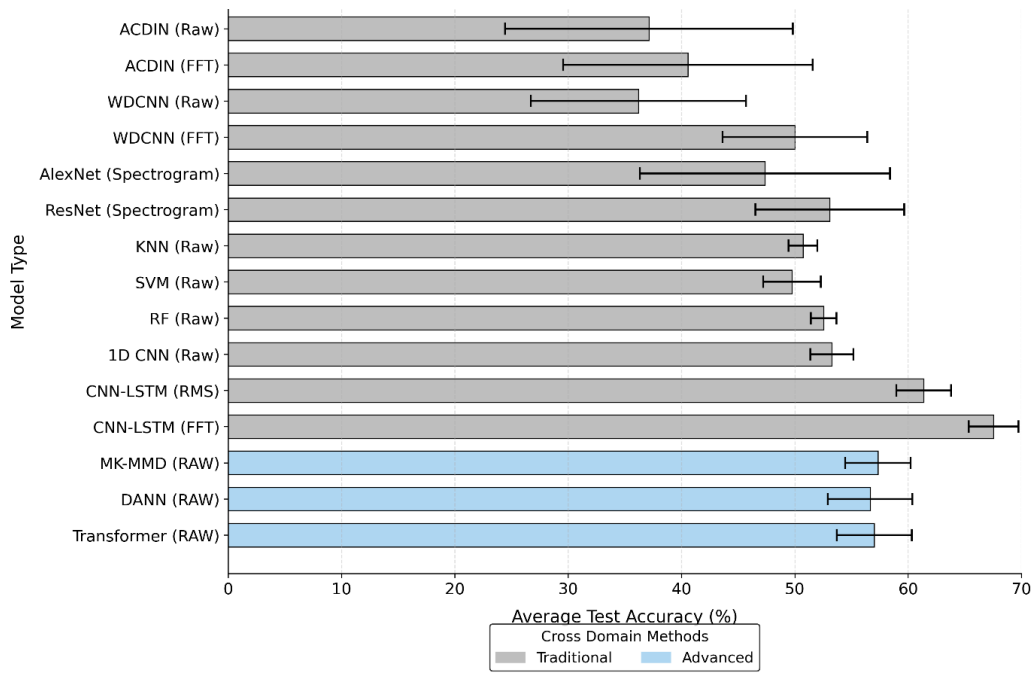


Figure 4-7. Average Test Accuracy When Splitting Data by Fault Severities with the CWRU Dataset

4.2.2 Triaxial Bearing Analysis

The input configuration for this section differs from Section 4.2.1. All experiments in this section use the triaxial bearing dataset collected at a sampling rate of 10,000 Hz. As in the previous section, raw time-domain accelerometer y-axis signals are segmented into fixed-length non-overlapping windows, with each window treated as one independent training sample. At this sampling rate, the window lengths correspond to the following physical durations: 512 samples = 51.2 ms, 1024 samples = 102.4 ms, and 2048 samples = 204.8 ms. These durations are approximately 4.8 times longer than the equivalent windows at 48,000 Hz, reflecting the lower sampling rate rather than a change in methodology. For SKF 6204 bearings at 1,700 RPM, the shaft frequency is 28.3 Hz and the BPFI is approximately 140.2 Hz. Therefore, a 1024 sample window at 10,000 Hz captures approximately 14.4 BPFI fault harmonics, providing slightly better fault cycle coverage than the equivalent window at 48,000 Hz. The same preprocessing methods (i.e., min-max, Z-score, RMS, and FFT) and the same CNN-LSTM architecture from Section 3.3 are applied without modification, allowing a direct comparison of generalization performance across the two datasets under consistent methodological conditions.

Table 4-5 highlights the impact of preprocessing, batch size, and input signal length on model generalization for the triaxial bearing dataset. While both 1D CNN and CNN-LSTM models achieve near-perfect training and validation accuracy, test accuracy remains lower, confirming that cross domain generalization is the key challenge rather than model capacity. Unlike the CWRU dataset, where the

CNN-LSTM outperforms the 1D CNN, the 1D CNN achieves the highest test accuracy on the triaxial bearing dataset, reaching $82.3\% \pm 2.6$ using FFT preprocessing, a batch size of 128, and an input signal length of 1024. The CNN-LSTM's best result of $65.8\% \pm 2.0$ is achieved using FFT preprocessing, a batch size of 128, and an input signal length of 1024, considerably lower than the 1D CNN's best result on this dataset. FFT preprocessing consistently outperforms min-max, Z-score, and RMS preprocessing across both architectures. These results reinforce the need for careful data preprocessing and structured data loading methods to bridge the gap between high training performance and reliable cross domain generalization.

Table 4-5 Accelerometer Results of Different ML Models for the Triaxial Bearing Dataset

Model Type	Preprocessing	Test Accuracy (%)	Batch Size	Input Signal
1D CNN	Min-max	59.8±2.0	64	512
		60.6±1.6	64	1024
		62.4±5.6	64	2048
		61.5±4.7	128	512
		64.7±2.6	128	1024
		63.5±2.6	128	2048
	Z-score	62.1±5.8	64	512
		63.5±7.1	64	1024
		67.6±2.6	64	2048
		61.0±2.5	128	512
		63.6±4.6	128	1024
		64.2±5.9	128	2048
	RMS	60.6±1.8	64	512
		62.6±3.6	64	1024
		70.5±3.9	64	2048
		63.4±2.1	128	512
		65.4±2.3	128	1024
		66.2±3.4	128	2048
	FFT	74.1±11.8	64	512
		76.7±2.8	64	1024
		78.9±2.6	64	2048
		65.9±3.3	128	512
		82.3±2.6	128	1024
		79.7±3.5	128	2048
1D CNN-LSTM	Min-max	51.1±2.4	64	512
		53.2±2.2	64	1024
		55.9±3.6	64	2048
		52.0±2.9	128	512
		55.1±2.0	128	1024
		56.9±3.6	128	2048
	Z-score	55.2±1.1	64	512
		56.2±1.6	64	1024
		57.6±4.9	64	2048
		55.1±1.3	128	512
		59.2±2.5	128	1024
		58.4±2.8	128	2048
	RMS	54.6±2.0	64	512
		60.8±2.0	64	1024
		59.5±5.6	64	2048
		60.8±4.8	128	512
		62.6±2.1	128	1024
		61.2±3.0	128	2048
	FFT	57.8±2.0	64	512
		62.1±2.5	64	1024
		60.3±4.1	64	2048
		61.8±4.3	128	512
		65.8±2.0	128	1024
		63.9±2.8	128	2048

Note: Training and validation accuracy exceeded 95% across all model configurations, indicating sufficient model capacity. Therefore, test accuracy is the primary metric for cross domain generalization.

4.2.3 UORED-VAFCLS Analysis

Table 4-6 demonstrates the performance of the 1D CNN and 1D CNN-LSTM architectures when used with the UORED-VAFCLS dataset’s accelerometer data. The 1D CNN model achieves the highest test accuracy overall, reaching $80.4\% \pm 4.0$ using FFT preprocessing, a batch size of 128, and an input signal length of 1024, with a second-best result of $76.7\% \pm 4.3$ at an input signal length of 512 under the same batch size and preprocessing conditions. In contrast, the 1D CNN-LSTM model’s best performance was found to be $75.6\% \pm 3.0$, achieved using FFT preprocessing, a batch size of 128, and an input signal length of 1024. Across both architectures, FFT preprocessing consistently outperforms min-max, Z-score, and RMS preprocessing. This is consistent with the patterns observed for the CWRU and triaxial bearing datasets in Sections 4.2.1 and 4.2.2. Like the triaxial dataset, the 1D CNN outperforms the 1D CNN-LSTM as the top performing architecture in this case.

Table 4-6 Accelerometer Results of Different ML Models for the UORED-VAFCLS Bearing Dataset

Model Type	Preprocessing	Test Accuracy (%)	Batch Size	Input Signal
1D CNN	Min-max	55.3±3.9	64	512
		59.9±4.8	64	1024
		58.9±5.3	64	2048
		63.3±4.8	128	512
		60.8±4.2	128	1024
		61.8±4.0	128	2048
	Z-score	60.8±6.4	64	512
		59.6±5.8	64	1024
		61.6±6.5	64	2048
		63.8±6.2	128	512
		61.8±6.0	128	1024
		59.2±6.3	128	2048
	RMS	63.9±6.3	64	512
		65.3±5.9	64	1024
		68.9±5.0	64	2048
		74.6±5.2	128	512
		71.3±5.9	128	1024
		70.0±4.8	128	2048
	FFT	65.5±6.9	64	512
		68.3±4.8	64	1024
		73.3±4.2	64	2048
		76.7±4.3	128	512
		80.4±4.0	128	1024
		75.6±4.9	128	2048
1D CNN-LSTM	Min-max	61.3±3.4	64	512
		64.5±4.8	64	1024
		63.8±4.9	64	2048
		65.9±4.8	128	512
		68.1±4.6	128	1024
		66.7±4.9	128	2048
	Z-score	63.4±3.6	64	512
		64.5±4.0	64	1024
		65.8±4.2	64	2048
		63.9±4.9	128	512
		60.2±4.3	128	1024
		63.9±4.1	128	2048
	RMS	64.6±3.2	64	512
		65.8±3.0	64	1024
		64.9±3.5	64	2048
		66.8±3.9	128	512
		69.9±3.0	128	1024
		59.8±3.3	128	2048
	FFT	65.8±3.1	64	512
		67.7±3.3	64	1024
		68.9±3.6	64	2048
		71.5±3.2	128	512
		75.6±3.0	128	1024
		70.8±3.0	128	2048

4.2.4 Discussion

Cross domain results across both the CWRU and triaxial bearing datasets reveal that even under simplified, controlled laboratory conditions, test accuracies are low for most configurations. Since these datasets contain seeded faults collected under clean, low-noise laboratory conditions, they should, in principle, be easier to generalize across than uncontrolled industrial data. Instead, existing public datasets were not designed with cross domain generalization in mind, and both datasets exhibit structural limitations that restrict the diversity and independence of domains available for evaluation.

The CWRU dataset relies exclusively on seeded faults and does not provide enough healthy bearing files to form an independent domain equivalent in size to the fault domains, limiting the range of health states available for cross domain evaluation. Similarly, the triaxial dataset does not include ball or cage fault conditions, restricting its use to inner race and outer race faults alone. Moreover, as discussed in Section 3.1.1.2 for the triaxial bearing dataset, the same physical bearing is recorded across all three load conditions rather than using physically distinct bearings per domain, meaning the operating condition (load) is the only true axis of separation between training and test domains rather than bearing identity. These structural constraints, rather than any ambiguity in the labeling of individual files, limit the diversity and independence of domains available for robust cross domain evaluation in both datasets.

These limitations motivated the design of a new dataset (UORED-VAFCLS) incorporating naturally developed faults, physically distinct bearings across every domain and health state, sensor placements chosen specifically to produce cleaner fault signals, and synchronized accelerometer and acoustic sensors for data collection. Acoustic data provides a second form of time-based frequency data that is distinct from the accelerometer data typically available in other datasets including the CWRU and triaxial dataset. Synchronisation of these two sensor types allows for the assessment of multi-source data loading methods.

The results in Sections 4.2.1 through 4.2.3 show that the 1D CNN-LSTM architecture is not consistently the top performer, achieving the best result on the CWRU dataset but being outperformed by the simpler 1D CNN on both the triaxial and UORED-VAFCLS datasets. A closer examination of why architecture performance varies across datasets points to two interacting factors, including window duration relative to the fault period, and the regularity of the fault signature itself. As shown in Section 4.2.1, the 1024-sample CWRU window at 48,000 Hz spans only about 3.5 BPFI cycles, and it is this dataset where the CNN-LSTM's temporal modelling provides its largest benefit over the CNN alone. This is consistent with a window that is too short to contain many repeated fault impacts, so learning how those impacts evolve within the sequence, rather than only their spectral content, becomes more informative. The triaxial dataset shows the opposite pattern. As indicated in Section 4.2.2, its 1024-sample window at 10,000 Hz already spans about 14.4 BPFI cycles, so the spectral content alone likely already captures most of the

periodic fault structure, leaving comparatively little additional information for the LSTM layer to extract sequence ordering from. Additionally, the added model complexity appears instead to increase the risk of overfitting to this dataset’s more limited domain diversity, since the triaxial cross domain splits are drawn from the same physical bearing across only three load conditions (Section 3.4), rather than from independent bearings.

Nonetheless, window duration alone does not explain the UORED-VAFCLS result. At the 42,000 Hz sampling rate used for this dataset, a 1024-sample window spans, again, only about 3.5 BPFI cycles, nearly identical to the CWRU window, yet the CNN outperforms the CNN-LSTM here as well. This discrepancy suggests that fault regularity, in addition to window duration, plays a role. The CWRU and triaxial datasets both contain seeded faults with a fixed, machined defect geometry, producing a repeatable, near-periodic impact train that a recurrent layer can learn to track consistently across domains. UORED-VAFCLS faults, in contrast, are allowed to develop naturally through each bearing’s own degradation process (Section 4.1), producing impact timing and amplitude values that vary stochastically both within and between bearings. Under a short window, this irregularity may be reflected in the sequence itself. Therefore, the LSTM layer may be learning transient, bearing-specific degradation patterns that do not transfer to unseen bearings, rather than a generalizable temporal signature, which would explain why the added recurrent capacity fails to help despite the short window duration that helped fault diagnosis performance when using the CWRU dataset.

Taken together, these results suggest that the CNN-LSTM’s temporal modelling is most beneficial when the fault signal within a window is both under-sampled in cycles and regular in structure. When either condition is not met, whether because the window already contains many repeated cycles, as with the triaxial dataset, or because the underlying fault process is naturally irregular, as with UORED-VAFCLS, the additional model complexity does not translate into improved cross domain generalization. This interpretation is supported by the window-length ablation study performed in Sections 4.2.1 through 4.2.3. Holding the batch size at 128 and FFT preprocessing as fixed, the advantage of the CNN-LSTM architecture on the CWRU dataset and the CNN architecture on the triaxial and UORED-VAFCLS datasets holds consistently across all three tested window lengths (512, 1024, and 2048 samples), rather than reversing as cycle coverage changes within a given dataset. Notably, at a 2048-sample window the CWRU and UORED-VAFCLS datasets span nearly identical BPFI cycle counts (approximately 6.9 and 7.0 cycles, respectively), yet the architecture preference diverges between them, indicating that fault consistency, not window duration alone, accounts for the difference.

Nonetheless, FFT preprocessing, a batch size of 128, and an input signal length of 1024 are consistently the best performing configuration across all three datasets regardless of which architecture is used. Therefore, these parameters and configurations are adopted for the remainder of this thesis. Still, the inconsistency in architecture performance motivates the investigation into data loading methods,

rather than model architecture, as the primary lever for improving cross domain generalization in Chapter 5.

Chapter 5

Data Loading Methods

This chapter introduces data loading methods and differentiates between data loading and data augmentation. Traditionally, data augmentation methods have been widely used for fault diagnosis, while data loading methods are scarcely researched. Data augmentation methods have been explored to artificially increase the size of training sets by including additive noise [62], amplitude scaling [63] and time shifting [64]. For example, additive Gaussian noise has been used to simulate sensor measurement uncertainty, amplitude scaling has been applied to mimic varying load conditions, and time shifting has been used to account for misaligned fault impact timing across samples. Unfortunately, increasing dataset size also increases computational requirements. Yet, these methods do not address how existing data should be structured before being presented to a model [65]. Importantly, data loading methods modify the order, source and composition of inputs seen by the algorithm at each optimization step without affecting the signals themselves [66]. These methods affect feature learning, convergence behaviour and cross domain robustness when data is limited, but they are underexplored in the bearing fault diagnosis literature.

5.1 Traditional Data Loading

Data loading refers to the process by which raw sensor data is segmented, organized, and presented to a ML algorithm during training, validation, and testing. While much of the existing literature focuses on model architecture and preprocessing methods, the way data is loaded into a model has an equally significant impact on the model’s ability to generalize across domains. Data loading determines the order, source and structure of inputs seen by the algorithm at each optimization step, directly influencing what features the model learns and how robust those features are to domain shifts.

In traditional data loading, raw time domain and other preprocessed signals are segmented into fixed length non-overlapping windows and randomly shuffled before being fed into the model. When data is drawn from a single sensor or source file, consecutive windows often share similar statistical properties, introducing short lag redundancy into the training. This redundancy reduces the effective diversity of information seen by the model per optimization step, increasing the risk of overfitting to source specific features rather than learning generalizable fault patterns.

Therefore, the choice of data loading method acts as a form of regularization. A loading method that maximizes the diversity of inputs seen by the model at each step helps with the learning of features that are consistent across sources and operating conditions, improving cross domain generalization without requiring additional data or a complex model architecture.

Specifically, when multiple sensors are available, alternating between their outputs within a single channel input can exploit source diversity without the computational overhead of parallel multi-channel loading. This principle motivates the data loading method proposed in Section 5.2, which addresses the limitations of traditional loading methods by systematically alternating between multiple data sources within a single channel input.

5.2 Selective Embedding

Dataset loading is often overlooked as a mechanism for improving generalization across machine health states even though inputs have a great effect on the output and model performance. To address this, a novel method for data loading is proposed herein, named selective embedding, which utilizes a diversity of data taken from the alternation of multiple sensors arranged into a single-channel input. The proposed method is specifically tailored to time domain data for enhancing generalization and improving model robustness under nonstationary conditions or dissimilar domains. When using selective embedding, instead of using all available data from a single source as in traditional loading, a portion of data is taken from each available source and combined into a single channel input, keeping the total amount of data that is fed into the model the same as in the traditional single channel method. A description of the proposed method compared to existing data loading methods is shown in Figure 5-1.

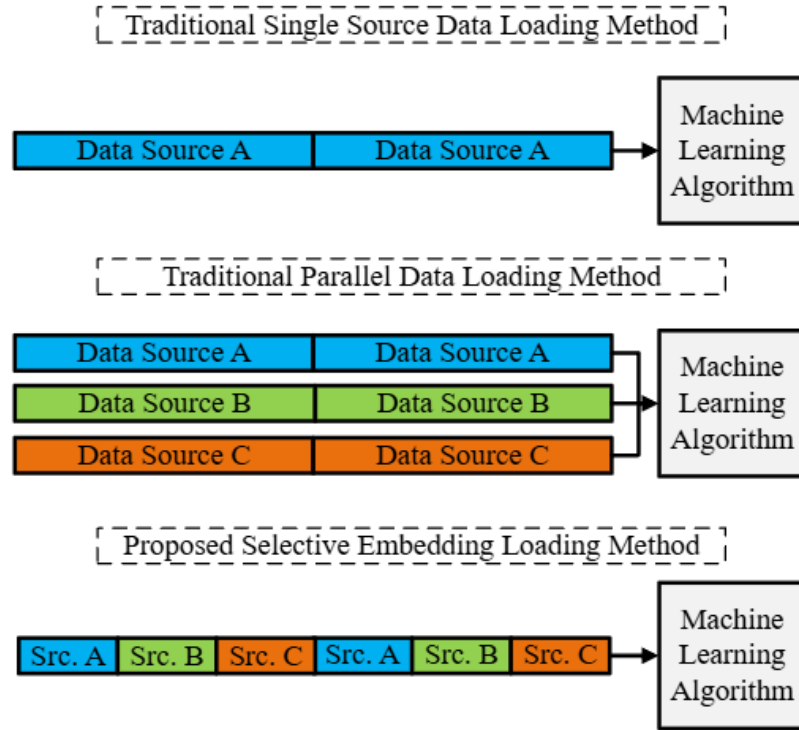


Figure 5-1. Comparison of Proposed Data Loading Method versus Existing

5.2.1 Method Overview

Current data loading methods are limited (i.e., researchers typically preprocess data, then select a single-axis input or load multi-source data in parallel channels) for training DL algorithms. For instance, the analysis in Chapter 4 is an example of single-axis data loading. Single data source-based methods are fast, but trained models suffer from a lack of diversity in the learned features. On the other hand, multi-source data loaded in a single channel sequentially or via parallel channels increases test accuracies but also increases the computation time or computational resources required to achieve this performance, respectively.

The proposed approach to data loading maintains or improves performance with a fewer number of diverse data when compared to traditional data loading methods, significantly improving computational efficiency. This chapter demonstrates the improved efficiency of the method by running experiments on several publicly available datasets and shows the applicability of this data loading method for different time domain scenarios.

The proposed input data loading method alternates between sources of data for all phases of model training, validation, and testing within a single channel input, rather than using the same source throughout or multiple channels. By alternating sources of data (Figure 5-2), the approach provides diverse perspectives on the data characteristics of the signals of interest, allowing the model to learn from a broader range of data, reducing the risk of overfitting.

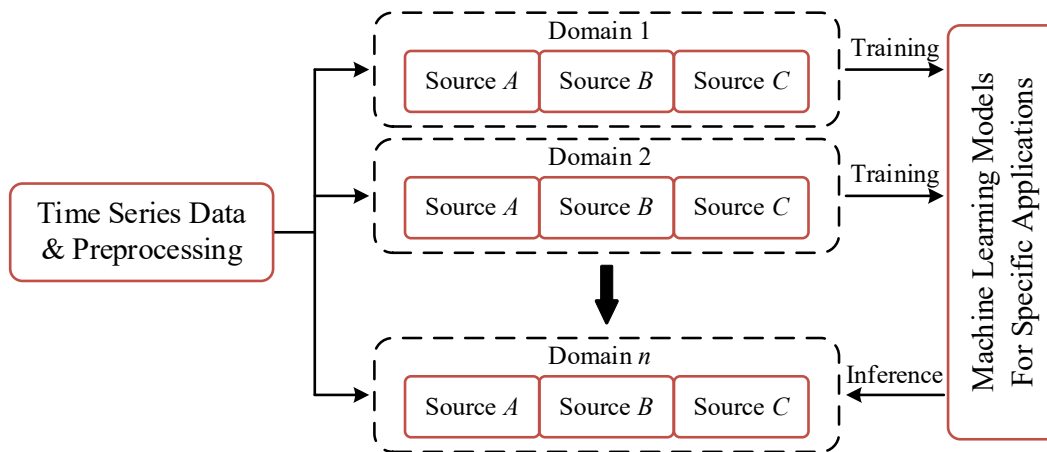


Figure 5-2. Selective Embedding Input Data Loading Across Different Sources

All objects have a fundamental resonant frequency that can be used as a feature in DL algorithms. To demonstrate the potential of selective embedding, time domain

data is converted to the frequency domain for all datasets used herein and analyzed using simple DL algorithms.

The Fourier spectrum is especially useful for feature extraction in frequency data classification, as it is robust against external noise and changes in environmental conditions. For example, in heavy machinery, variations in load may affect the amplitudes of frequencies for different objects or object conditions, the characteristic fault frequencies of these machines and their frequency patterns remain constant. This consistency makes the Fourier spectrum a powerful tool for identifying any object that has a resonant frequency.

When each data source is loaded using selective embedding, the frequency components used for classification remain consistent while providing a wide range of amplitude variations across different conditions, allowing the algorithm to differentiate between features more effectively. Classification refers to a technique where the ML model categorizes a set of data points (class) based on features using a labeled dataset to learn relationships between features of classes. This single-channel multi-data source input method uniquely diversifies the data fed into the model, capturing the inherent variability in a system's behavior and improving overall performance while reducing computational requirements due to the smaller size of training data. This proposed method leverages the strength of selective embedding to capture frequency domain representations, ensuring that critical frequency-specific features, like those associated with mechanical faults, are effectively learned. However, the choice of dataset domains must be considered before processing the

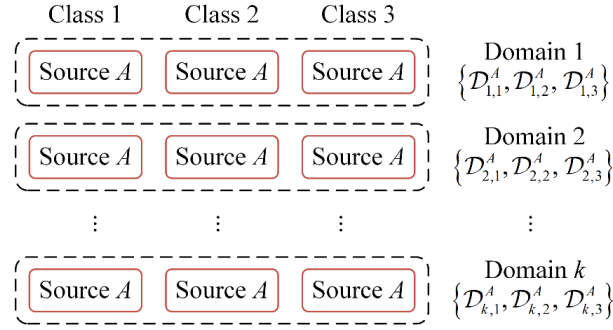
data. To quickly recap, a domain refers to a set of data files labeled with different classes created to understand the features, model architecture, and interpretation of results. Domain selection is intended to enhance the generalizability of object classification. Domain splitting refers to the careful creation of domains where each domain consists of a dataset from a different instance to ensure that data fed into training, validation, and testing preserves diversity while preventing data leakage between domains. This structured division strengthens the learning process and contributes to the consistent and repeatable performance of the proposed method across datasets.

An important methodological distinction between data augmentation and data loading methods is highlighted in Section 2.2.2. This distinction is intentional and justified by the nature of selective embedding itself. As established in Section 4.2, a FFT frequency domain representation is used because characteristic fault frequencies remain consistent across sources and operating conditions regardless of amplitude variations, making it the most suitable input for demonstrating the benefits of alternating between multiple data sources. Therefore, the goal of Chapter 5 is not to extend the absolute accuracy comparison of Chapter 4 but to evaluate the relative improvement. Each 1024-sample time-domain window is transformed into a 1024-point FFT magnitude spectrum, producing a 1D frequency-domain tensor of shape (1×1024) where each element represents the signal energy at a specific frequency bin. This representation is used consistently across all six datasets in this chapter regardless of sensor type or industry. The frequency resolution of each bin

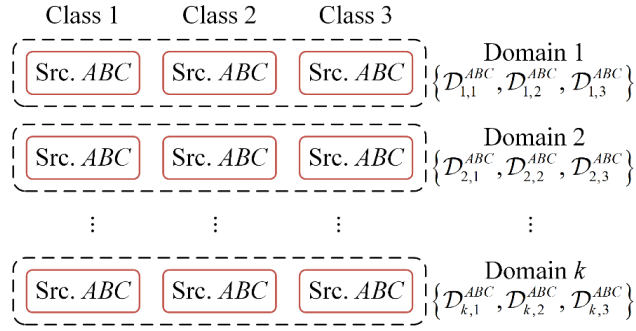
depends on the dataset sampling rate and is summarized as follows, Dataset 1 at 6,400 Hz gives 6 Hz/bin; Dataset 2 at 10,000 Hz gives 9 Hz/bin; Dataset 3 at 12,000 Hz gives 11 Hz/bin; Dataset 4 at 42,000 Hz gives 41 Hz/bin; Dataset 5 at 42,000 Hz gives 41 Hz/bin; Dataset 6 at 64,000 Hz gives 62 Hz/bin. While Dataset 6 has a coarser resolution of 62 Hz/bin due to its higher sampling rate, the fault frequencies of interest for the train transmission system remain separable within this resolution, and the CNN learns from the overall spectral pattern rather than requiring fine grained frequency precision. The motivation for this switch from time-domain to frequency-domain representation has three reasons. First, as established in Section 2.1.1, bearing fault frequencies and their harmonics are stable in the frequency domain regardless of amplitude variations caused by load, sensor position, or operating condition, making frequency-domain features more transferable across domains than raw time-domain amplitudes. Second, selective embedding alternates between multiple sensor sources whose time-domain amplitudes differ substantially due to sensor type or placement; the FFT normalizes these amplitude differences by focusing on frequency content rather than raw signal level. Third, the six datasets in this chapter span multiple industries and sensor configurations, and the frequency domain provides a common representational basis that allows for a fair cross-dataset comparison. Therefore, Chapter 4 results assess single channel time-domain and frequency-domain data baseline performances on existing benchmark datasets, while Chapter 5 results characterize the selective embedding method under preprocessed frequency-domain inputs. Chapter 4 dictates that frequency domain has a baseline

performance that is superior to time domain analysis and Chapter 5 demonstrates the gains achievable through different data loading methods and compares results to those obtained using the proposed selective embedding data loading method under frequency domain inputs.

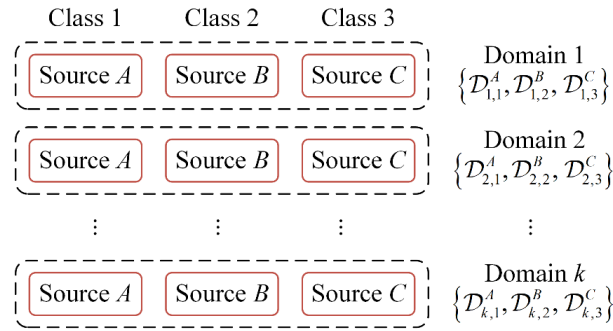
Figure 5-3 compares three different data loading methods. In (a) traditional single-channel loading, only one source (e.g., Source A) is used for all classes across domains, which limits data diversity and generalization. In (b) traditional parallel-channel loading, each source (A, B, C) is loaded in a parallel channel, which increases diversity but also requires greater computational resources. In contrast, (c) selective embedding alternates short segments from different sources (A, B, C) within the same input channel. This maintains diversity while avoiding the high computational costs of parallel loading. This is achieved by using only a portion of the available data from each source, keeping the total amount of data equivalent to a traditional single channel approach while still benefiting from the diversity of multiple sources. For each domain, raw sensor data is divided into fixed-length, non-overlapping windows (1024, FFT-processed signals), where the 1024 sample window length is selected based on the empirical results of Chapter 4. This method ensures that the model sees representative information from all sources on an ongoing basis while maintaining computational efficiency.



(a) Traditional loading (single)



(b) Traditional loading (parallel)



(c) Proposed loading (selective embedding)

Figure 5-3. Comparison of input data loading methods: (a) traditional single channel, (b) traditional parallel multi-channel, and (c) proposed selective embedding with alternating sources.

The algorithmic steps for implementing selective embedding are presented in Table 5-1, which outlines how the method systematically alternates data sources during training and testing across domains. The pseudocode provided in Table 5-1

illustrates how selective embedding cyclically alternates between data sources for each condition during training (lines 5–9) and for each sample during testing (lines 13–17). Instead of combining data from all sources simultaneously, the algorithm rotates through the available sources using modular indexing, ensuring that each condition or test sample is loaded from a different source in a systematic and alternating fashion. This method helps introduce diversity in the data, as seen by the model, improving generalization and reducing overfitting across domains.

Table 5-1 Algorithm 1: pseudocode for selective embedding.

Algorithm 1 Selective Embedding	
Require: Dataset $\mathcal{D}$ with C conditions, k domains, and S sources	
Ensure: Performance estimate across domains	
1:	for each domain $d_{test} \in \{1, \dots, k\}$ do
2:	$\mathcal{D}_{train} \leftarrow \mathcal{D} \setminus \mathcal{D}_{d_{test}}$
3:	$\mathcal{D}_{test} \leftarrow \mathcal{D}_{d_{test}}$
4:	Initialize empty training set $\mathcal{T}$
5:	for each condition $c \in \{1, \dots, C\}$ do
6:	$s_c \leftarrow (c \bmod S) + 1$ $\triangleright$ Rotate sources cyclically
7:	Load samples from $\mathcal{D}_{train}$ with condition c and source s_c
8:	Add loaded samples to training set $\mathcal{T}$
9:	end for
10:	Train classifier f using $\mathcal{T}$
11:	Initialize empty test set $\mathcal{T}_{test}$
12:	$i \leftarrow 0$
13:	for each sample $x \in \mathcal{D}_{test}$ do
14:	$s_i \leftarrow (i \bmod S) + 1$
15:	Load sample x from source s_i
16:	Add $(x, label)$ to test set $\mathcal{T}_{test}$
17:	$i \leftarrow i + 1$
18:	end for
19:	Evaluate f on $\mathcal{T}_{test}$ and record performance
20:	end for
21:	return Aggregated performance across all domains

5.2.2 Experiment Setup

To demonstrate the use of selective embedding, heavy machinery, railroad, and manufacturing datasets are selected, as discussed in Section 3.1. The 6 datasets are selected to establish the superiority of selective embedding when compared to traditional methods.

Dataset 3 (CWRU) is used in this section to demonstrate how time domain data is preprocessed using the FFT for training DL models. The time domain signals in Figure 5-4 a) illustrate vibration data across domains 0, 3, and 6 (inner race fault) for different fault severities. Even though the amplitudes vary between domains due to differences in fault sizes and sensor positions, the periodic nature of the signal remains evident. Important features for similar fault types also remain the same. Selective embedding helps distinguish these important similarities, improving the ability of DL models to generalize patterns by reducing input complexity but maintaining information diversity.

In the frequency domain (Figure 5-4 b), the FFT-transformed signals show that the dominant frequency components across domains match the fault frequencies expected for the operating conditions at hand, which is true for any object. However, the amplitudes at these frequencies differ between fault types, which provides critical information for fault classification. Other important features also remain similar. This consistent frequency response combined with amplitude variations ensures that selective embedding maintains robust fault characterization.

Selective embedding effectively leverages a reduced portion of the original dataset by eliminating portions of data from every source during data loading in an alternating sequence. For example, in the CWRU 12 kHz bearing dataset, where each file contains 120,000 samples, the data is split into smaller, non-overlapping segments of 1024 FFT preprocessed samples (i.e., frequency bins) coming from both the drive end (DE) and fan end (FE) sensors. These input segments are then

alternated between DE and FE during training, validation, and testing. This allows for significant data reduction while retaining the critical features needed for condition classification, improving computational efficiency and maintaining a high accuracy performance.

Two clarifications are required for the CWRU dataset usage in this chapter. First, the 12,000 Hz drive end data is used here rather than the 48,000 Hz data used in Chapter 4. This choice is intentional. At 12,000 Hz, a 1024-sample FFT window provides a frequency resolution of 11 Hz/bin and captures up to 6,000 Hz, which is sufficient to resolve the BPFI at approximately 162 Hz and its first 37 harmonics for the SKF 6205 bearing at 1,797 RPM, well above the 10-harmonic minimum established in Section 2.1.1. The 12,000 Hz dataset also provides the FE accelerometer channel in addition to the DE channel, allowing the two-source alternation that selective embedding requires. The 48,000 Hz dataset does not provide a second synchronous sensor channel suitable for this alternation, making it unsuitable for the selective embedding framework as implemented here.

Figure 5-4 illustrates this process using three different domains from the CWRU dataset with varying fault severities (Domain 0: DE, 0.007 in; Domain 3: FE, 0.014 in; Domain 6: DE, 0.021 in). The time-domain signals show similar waveform structures but varying amplitudes due to sensor type and fault size, while the FFT-transformed signals display consistent peak frequencies corresponding to bearing fault harmonics. Amplitude variations across domains provide critical information

for classification. In this case, the red dashed lines in the spectra mark known BPFI harmonics, confirming spectral alignment.

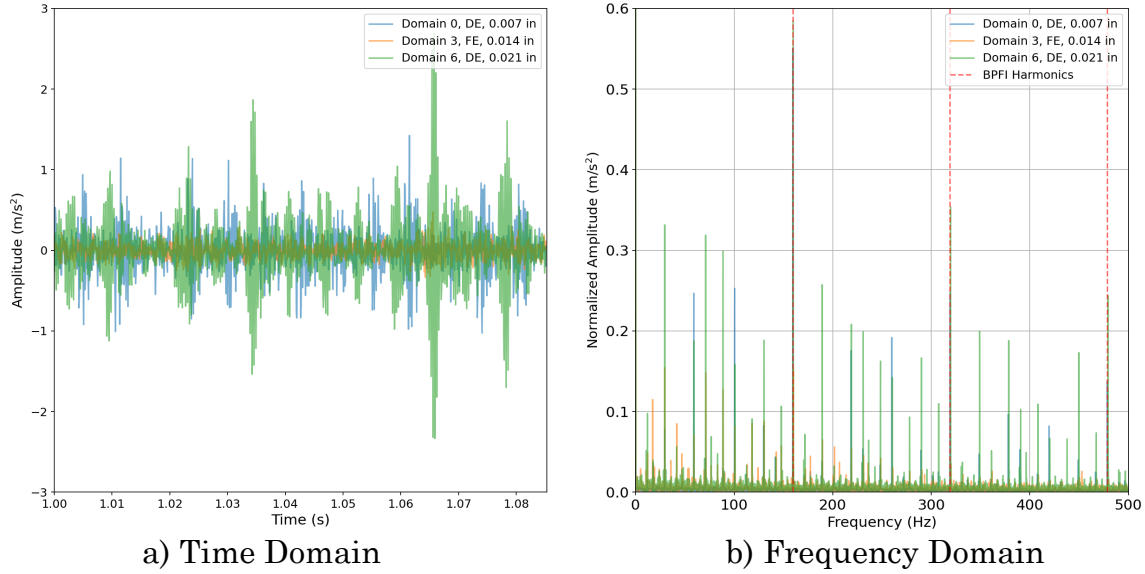
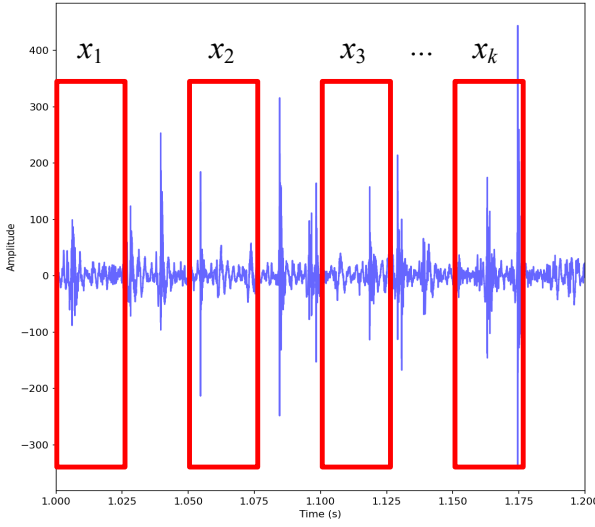


Figure 5-4. CWRU dataset domains showing time- and frequency-domain signals with varying fault severities.

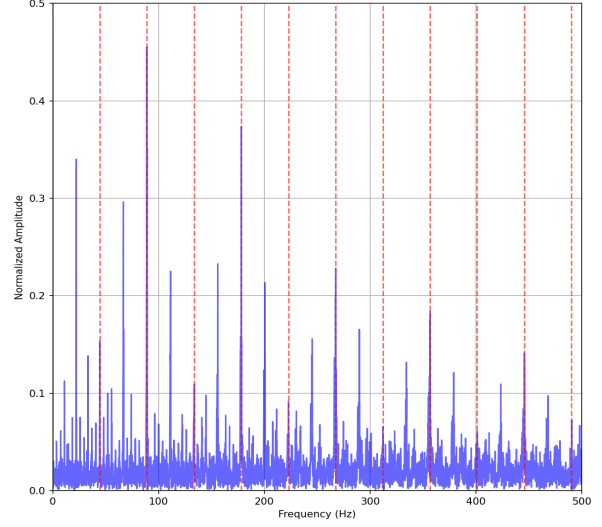
Figure 5-5 further demonstrates selective embedding where time domain data segments (i.e., time-signal bins) used as inputs are highlighted in red boxes for both accelerometer and acoustic data from dataset 4 (UORED-VAFCLS). The numbering of each box shows how each data segment is loaded into the same channel of the DL algorithm for feature extraction. This can be done because data is collected simultaneously (i.e., at the same time) from each sensor, ensuring that frequency data from all sensors overlap, as shown in Figure 5-4.

In Figure 5-5, the top row shows accelerometer signals, and the bottom row shows acoustic signals. Panels (a) and (c) display raw time-domain signals with red boxes marking the non-overlapping FFT windows from each sensor, while panels (b) and

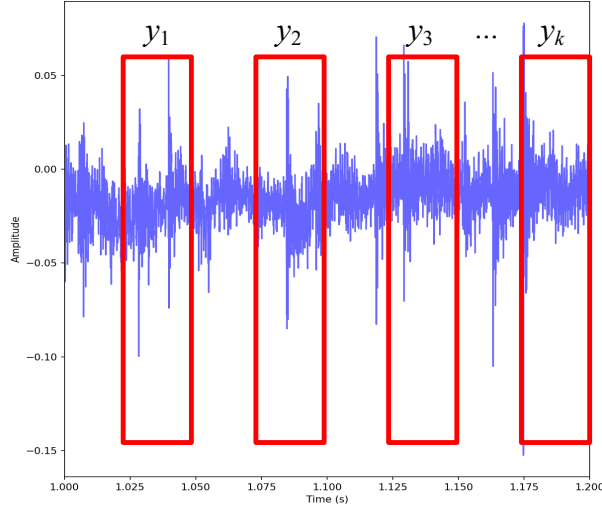
(d) show the corresponding frequency spectra. These segmented and transformed inputs are then alternated during data loading into a single-channel model, enhancing representation diversity and reducing overfitting.



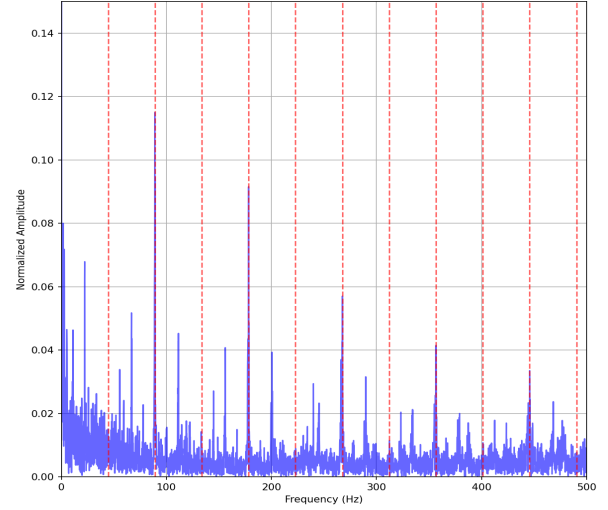
a) Accelerometer Data in the Time Domain



b) Fourier Spectrum of a)



c) Acoustic Data in the Time Domain



d) Fourier Spectrum of c)

Figure 5-5. Selective embedding with synchronized accelerometer and acoustic signals from the UORED-VAFCLS dataset [26].

Data contained within each red box of Figure 5-5 can be combined according to equation (11) and (12) to produce the data stream that is used as an input for the DL algorithm. Let values from the first data source $\mathbf{A}$ (e.g., accelerometer in this example) which represent an arbitrary 1024 signal length from Figure 5-5 a) be represented by

$x_1, x_2, x_3, \dots, x_k$, where each sample is a feature vector (e.g., $x_1=[x_{1,1}, x_{1,2}, x_{1,3}, \text{and } x_{1,1024}]$). Similarly, 1024 signal segments highlighted in red boxes from the other source **B** (e.g., acoustic sensor), as shown in Figure 5-5 c), are represented by $y_1, y_2, y_3, \dots, y_k$ (i.e., $y_2=[y_{2,1}, y_{2,2}, y_{2,3}, \text{and } y_{2,1024}]$). D represents the constructed dataset considering different class labels using the proposed selective embedding data loading approach, expressed as

$$D = \bigcup_{j=1}^{n_c} D_j \quad (11)$$

where for each class j with a label of l_j , the data sample selected are formulated as

$$D_j = \begin{cases} \{(x_i^{(j)}, l_j) \mid i = 1, 2, 3, \dots, k\}, & x_i^{(j)} \in \mathbf{A} \ (j = 1, 3, 5, \dots, j \leq n_c) \\ \{(y_i^{(j)}, l_j) \mid i = 1, 2, 3, \dots, k\}, & y_i^{(j)} \in \mathbf{B} \ (j = 2, 4, 6, \dots, j \leq n_c) \end{cases} \quad (12)$$

where $x_i^{(j)}$ denotes the i -th sample from class j in source **A**, $y_i^{(j)}$ denotes the i -th sample from class j in source **B** (**A** denotes accelerometer data while **B** denotes acoustic data).

Results for the six datasets included in this study are used as benchmarks when testing different DL architectures and indicate that selective embedding is effective and efficient with all networks.

After defining the architecture and hyperparameter settings in Section 3.3.2 for each of the baseline and proposed models, 10 independent experiments are performed for each task. Each experiment consists of one full training run from random weight initialization, using the same fixed domain split, hyperparameters, and preprocessing throughout. The only variable across the 10 runs is the random seed for weight

initialization and the stochastic elements of Stochastic Gradient Descent (SGD) which is an optimization algorithm used to iteratively update the network weights by computing gradients on small random subsets of the training data, called mini-batches, rather than the full dataset at once. Therefore, reported mean accuracy and standard deviation reflect initialization sensitivity under a fixed experimental configuration, not variability across different domain assignments. This approach ensures that performance differences between loading methods are attributable to the loading method itself rather than to favorable data splits.

5.2.3 Results and Discussion

The results in Table 5-2, Figure 5-6 and Figure 5-7 highlight how the proposed method provides a stable and efficient solution across diverse datasets. Unlike traditional approaches that either load a single source (single channel loading) or use multiple sources simultaneously (parallel channel loading), selective embedding switches between different data sources, enabling the benefits of multi-source information usually provided in parallel channels, while retaining the advantages of single-channel simplicity and efficiency. This balance improves generalization, avoids overfitting, and reduces computational demands.

While the traditional parallel loading method occasionally outperforms the proposed method with slightly higher accuracy (less than 3% difference for datasets 1, 5, and 6 when using a simple CNN), it comes at a significant computational cost. This computational cost difference is due to selective embedding preserving the same total data volume, batch size, and number of training iterations as traditional single-

channel loading, differing only in which source each segment is drawn from at each step. As a result, selective embedding requires no additional computational time relative to single-channel loading. This is unlike parallel loading, which combines data from all sources simultaneously and, therefore, processes a proportionally larger effective input at each step. For instance, when using the CNN with dataset 5, parallel loading provides an accuracy of 98.2% after 110 seconds while selective embedding provides a very good accuracy of 95.3% after only 20 seconds. Nonetheless, selective embedding provides higher accuracies for most cases when compared to other data input methods, but at significantly reduced model training times. This is especially true for more complex datasets, as can be seen with datasets 2, 3, and 4 in Table 5-2, and Figure 5-6.

Selective embedding outperforms traditional data loading methods. The key contribution of the proposed method lies in its efficiency and ability to stabilize all networks, significantly reducing sensitivity to noise. Across all datasets and algorithms tested, the method maintains over 80% accuracy (usually over 90%), even under challenging conditions like those in dataset 3, where traditional methods often fall short. Specifically, selective embedding achieves 84.8% (CNN) and 92.4% (CNN-Transformer) for dataset 3, a notable improvement over traditional data loading methods that remain below a 70% accuracy threshold.

The contribution of selective embedding is specifically demonstrated by the controlled comparisons in Table 5-2, dataset 3, where single-channel and parallel-channel loading methods use the same FFT frequency-domain input as selective

embedding. Selective embedding achieves 84.8% versus 50.3% for single-channel loading and 55.4% for parallel loading, confirming that the performance gain is attributable to the loading method rather than the input representation change alone.

Cases where the proposed method does not outperform traditional data loading methods only occur when using simple datasets (e.g., datasets 1 – few classes and dataset 5 – few instances of each class). This reduced diversity in the data limits the advantage of alternating between sources. This trade-off reflects the fundamental difference between the two methods: parallel loading uses all sources at every step, maximizing the information available per update at the cost of higher computation, whereas selective embedding provides one source at a time in alternation, trading some per-step information for single-channel computational efficiency. However, in more complex scenarios (e.g., dataset 3 or 4), where diversity in the data is high, selective embedding consistently and significantly outperforms other methods, demonstrating its strength for use in real-world applications where complexity and diversity are common.

The performance of selective embedding over traditional loading is strongly dependent on the number of classes and the variety of health states within each of the six datasets. The datasets with fewer classes and lower variety (e.g., Dataset 1 with 16 flow conditions considered as largely homogeneous states and Dataset 5 with a limited number of motor fault states) show the smallest performance gap between selective embedding and traditional loading, and in some cases traditional parallel

loading performs marginally better. In contrast, datasets with a richer variety of classes and more complex inter-class variations (e.g., Dataset 3 and Dataset 4, both containing multiple types and severity levels of faults) are the ones that gain the most benefit from selective embedding. This observation implies that the advantage of selective embedding depends on the number and variety of classes available for the method to leverage via source alternation: datasets with fewer distinct classes gain less from the additional diversity that source alternation provides, while datasets with a more complex class structure benefit more from the increased information diversity per iteration.

In summary, the novelty of selective embedding lies in its ability to integrate multiple data sources in an alternating manner while reducing the total amount of data used, thereby mitigating the risk of overfitting and the high computational costs commonly seen in traditional data loading methods. Moreover, its ability to stabilize and improve the reliability of these algorithms or datasets makes it particularly suitable for implementation in industry. Selective embedding does not always outperform other data loading methods when these methods perform well already. However, this only occurs with simple datasets, and in these cases, the performance of the proposed method is only slightly less than traditional loading methods. Nonetheless, for difficult tasks, where traditional data loading methods fail to provide adequate results, selective embedding maintains a high performance appropriate for industry. By providing stable and reproducible performance across various datasets

and algorithms, selective embedding highlights a practical pathway towards achieving more efficient and human-like data processing in DL applications.

Table 5-2 Comparison of data loading methods when using a CNN and a CNN-Transformer

	Single Channel Data Loading (Traditional)		Parallel Data Loading (Traditional)		Selective Embedding (Proposed)	
CNN	Test Accuracy (%)	Time (s)	Test Accuracy (%)	Time (s)	Test Accuracy (%)	Time (s)
Dataset 1	93.3±1.9	10	94.5±2.1	65	92.2±1.1	10
Dataset 2	82.3±2.6	10	86.7±2.0	65	95.6±0.3	10
Dataset 3	50.3±3.0	25	55.4±3.6	200	84.8±2.7	25
Dataset 4	80.4±4.0	15	84.6±4.8	135	94.2±5.6	15
Dataset 5	97.0±0.6	20	98.2±1.2	110	95.3±1.5	20
Dataset 6	98.7±1.6	60	99.9±0.1	110	97.6±4.9	60
CNN-Transformer	Test Accuracy (%)	Time (s)	Test Accuracy (%)	Time (s)	Test Accuracy (%)	Time (s)
Dataset 1	77.5±4.0	20	79.2±3.6	190	80.6±2.0	20
Dataset 2	90.6±5.5	20	92.7±4.5	270	95.9±1.0	20
Dataset 3	64.5±8.6	40	67.4±7.5	460	92.4±4.1	40
Dataset 4	78.1±4.4	35	82.5±4.0	280	96.5±1.5	35
Dataset 5	99.4±0.4	35	99.4±0.7	225	99.1±1.0	35
Dataset 6	99.3±0.6	120	99.6±0.5	140	98.6±2.6	120

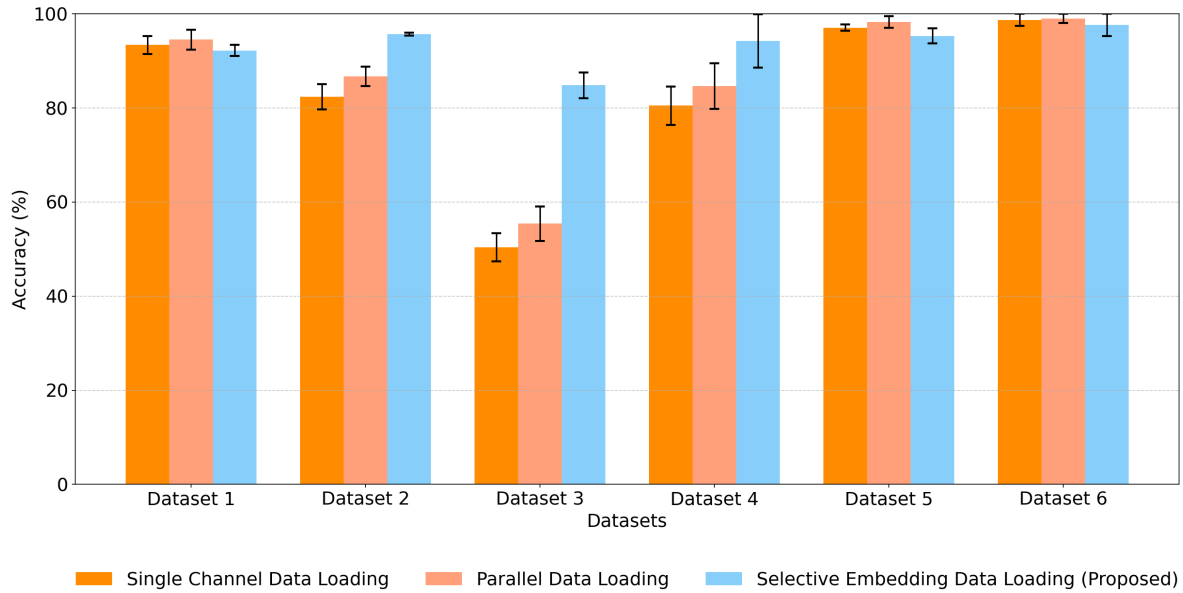


Figure 5-6. Test accuracy distribution for data loading methods for all datasets using a simple CNN architecture [26].

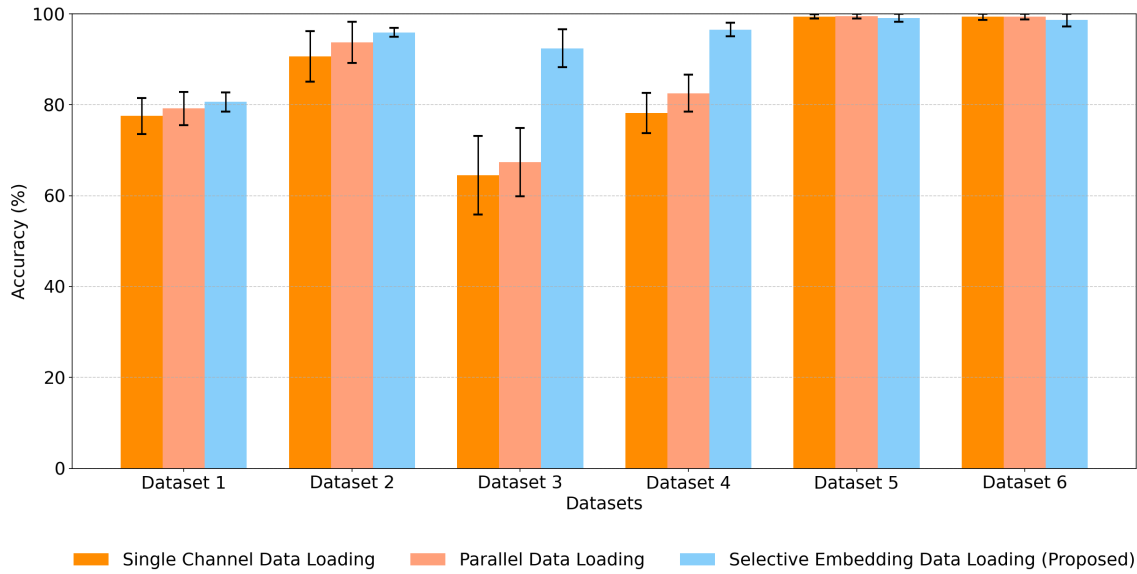


Figure 5-7. Test accuracy distribution for data loading methods for all datasets using a CNN-Transformer [26].

To further validate the performance of selective embedding, extended test results using five different DL architectures: CNN, RNN, GRU, CNN-Transformer, and ResNet18 are provided. Each model was evaluated across the six benchmark datasets using two data loading methods: multi-channel, parallel data loading and single-channel, multi-source data loading (selective embedding).

To visually compare the performance across datasets and architectures, Figure 5-8 and Figure 5-9 present the test accuracy distributions using radar plots for each respective model.

Across a majority of the cases, selective embedding consistently achieves higher test accuracy and lower variance, particularly on datasets 2, 3, and 4. These datasets are known to be more complex due to a higher diversity in the data. For these cases, selective embedding proves to be particularly effective, especially when paired with transformer-based architectures.

Importantly, even for simpler datasets (e.g., dataset 1 or 5), selective embedding performs well when compared to traditional data loading methods at a fraction of the computational cost. This highlights the method’s adaptability, and scalability, making it suitable for both complex and straightforward ML tasks involving multi-source data.

These results reinforce the practical benefits of selective embedding for real-world ML tasks involving diverse sources of data, complex operating environments, and time-series signals. The method maintains high accuracy across architectures and datasets while reducing computational time or requirements.

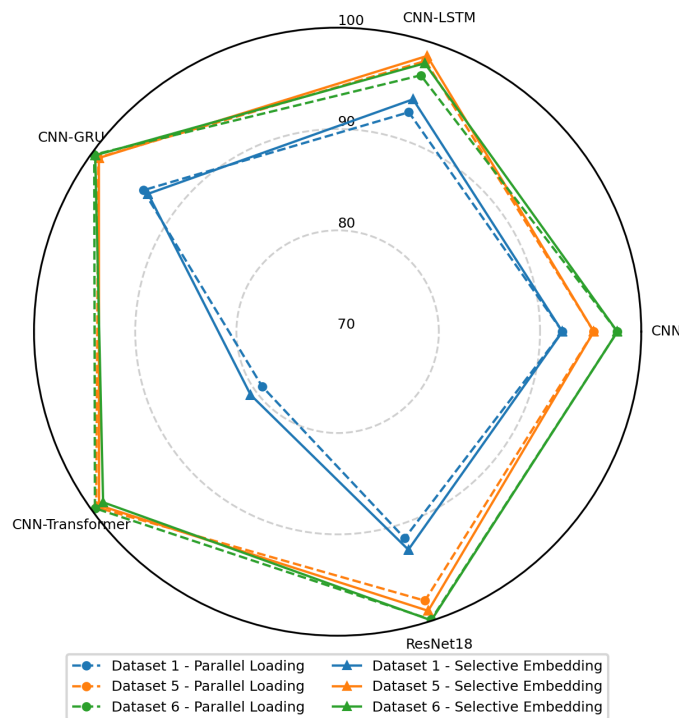


Figure 5-8. Radar plot using datasets 1, 5, and 6 for different ML algorithms [26].

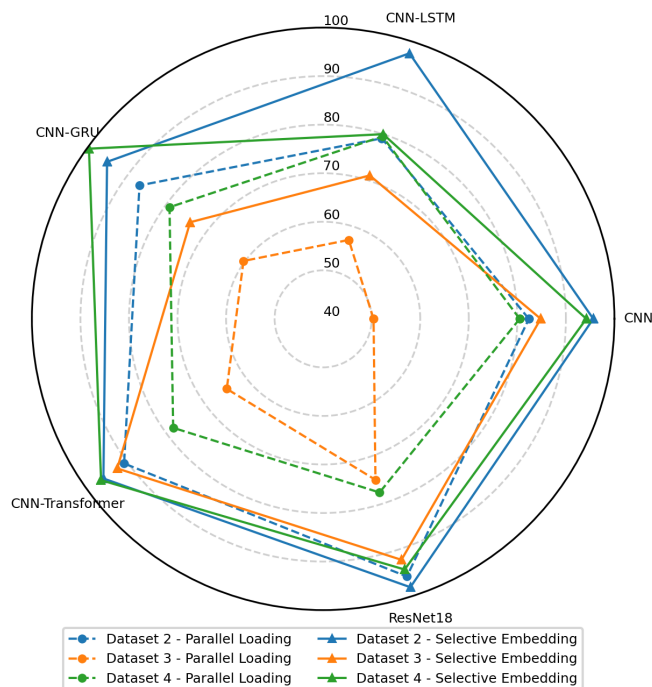


Figure 5-9. Radar plot using datasets 2, 3, and 4 for different ML algorithms [26].

The consistent performance gains of selective embedding across all datasets and architectures suggest that the benefit of alternating between data sources goes beyond simple data augmentation. Specifically, the alternation reduces redundancy in the training stream and increases the diversity of information seen by the model at each optimization step, a hypothesis that is formalized in Chapter 6.

Chapter 6

Theoretical Foundations of Selective Embedding

Following the empirical results presented in Chapter 5, this chapter provides the theoretical foundations for why selective embedding enhances generalization across domains. The main argument is that redundancy in the training data stream hurts model performance, while diversity helps. This chapter formalizes the notions of redundancy, diversity and generalization within the context of stochastic gradient descent and explains the mechanisms that lead to the accuracy improvements observed in Chapter 5. The notation introduced here is consistent with standard ML theory and is intended to complement rather than replace the practical understanding developed in the preceding chapters.

Selective embedding changes when and from where training data is loaded prior to preprocessing, without modifying the underlying content. By altering the input stream statistics seen by SGD, selective embedding increases the effective information per optimization step, reduces variance, and tightens generalization bounds under dependent sampling.

6.1 Problem Setup (Defining the Data and Domain)

To formalize how selective embedding works, the training structure must first be defined. Each domain $d \in \{1, \dots, D\}$ contains multiple sources $s \in S_d$, where each source represents a distinct sensor or data file. From each source, raw data segments $z_{d,s,i} \in Z$ are collected with labels $y_{d,s,i} \in Y$, consistent with the domain splitting approach used throughout this thesis. Where d is the domain, s is the source within that domain, and i indexes the individual sample. Each raw data segment z is preprocessed prior to being passed to the model, using one of the methods introduced in section 3.2. Chapter 4 showed that FFT preprocessing improves cross domain generalization by exposing frequency-domain fault signatures that remain more stable across operating conditions than raw time-domain amplitudes. Let $\mathcal{P}$ denote the preprocessing applied to each segment (e.g., min-max, Z-score, RMS, or FFT preprocessing signal methods evaluated in Chapter 4), as shown in equation (13):

$$x = \mathcal{P}(z). \quad (13)$$

Each preprocessed data segment x is passed through the model f_θ , parameterized by θ , the trainable parameters within the convolutional and fully connected layers described in Table 3-2, which outputs a predicted probability distribution $\hat{y}$ over the fault classes, computed as $\hat{y} = f_\theta(x)$. The goal of training is to minimize the expected loss over all domains and sources, as in equation (14)

$$\mathcal{L}(\theta) = \mathbb{E}_{(x,y)}[-\sum_{i=1}^C y_i \log(\hat{y}_i)], [67] \quad (14)$$

where C is the number of fault classes, y_i is 1 if the fault label y corresponds to class i and 0 otherwise (the encoded ground truth), and $\hat{y}_i$ is the model's predicted probability for class i , obtained by passing x through f_θ . Let $\ell(f_\theta(x), y) = -\sum_{i=1}^C y_i \log(\hat{y}_i)$ denote the per-example loss, so that the total $\mathcal{L}(\theta) = \mathbb{E}_{(x,y)}[\ell(f_\theta(x), y)]$. $\mathbb{E}_{(x,y)}$ represents the average of the per-example loss over every training example provided, where x is a preprocessed data segment and y is its corresponding fault label, provided as a pair since y is x 's label, averaged across all domains and sources.

6.2 Redundancy Versus Diversity

The word diversity used throughout this chapter refers to the variety of distinct sources (e.g., sensors, operating conditions) contributing samples to the training stream, not to the difficulty or complexity of the underlying classification task. A dataset can be diverse in its sources while remaining a comparatively simple classification problem, or complex in its class structure while being drawn from a single, non-diverse source. This distinction matters because increased task complexity (e.g., highly confusable classes, high intra-class variability) is known to degrade model performance, whereas the increased source diversity introduced by selective embedding is shown in this chapter to improve generalization by reducing redundancy in the training stream.

In traditional single channel loading, samples are drawn from one source in sequence, which creates redundancy across consecutive samples since back-to-back windows from the same file share similar statistical properties. This redundancy is measured as a correlation across consecutive samples, while diversity is measured by the entropy rate of the training stream. High redundancy reduces the effective amount of useful information seen by the model per optimization step and inflates variance in SGD updates, while higher diversity increases the information carried per step, leading to better generalization.

6.3 What Selective Embedding Changes

Rather than drawing all samples from a single source, selective embedding introduces an alternation schedule $\sigma(t)$ that switches between sources before preprocessing, as in equation (15):

$$x_t = \mathcal{P}(z_{\sigma(t)}). \quad (15)$$

Here, t denotes the training step (SGD iteration), and $\sigma(t)$ is the alternation schedule that selects the source used at step t . Where x_t depicts the preprocessed input obtained by applying $\mathcal{P}$ to segment $z_{\sigma(t)}$, drawn from the alternated source selected at step t . As a result, each SGD update is based on inputs from different sources, increasing diversity per step and reducing local redundancy. This leads to lower gradient variance across the batch, formally defined in equation (16) as

$$\text{Var}(\nabla_{\theta} \ell(f_{\theta}(x_t), y_t)). \quad (16)$$

where ∇_{θ} is the gradient with respect to the model parameters θ , so $\nabla_{\theta}\ell(f_{\theta}(x_t), y_t)$ is the per-example gradient of the loss at step t , evaluated using x_t and its corresponding label y_t , the label of segment $z_{\sigma(t)}$, and Var denotes the variance of this gradient taken across the batch.

6.4 Theoretical Information

To quantify the information content of the training stream, let $\{x_t\}$ represent the sequence of inputs seen by the model during training. Here, X is the stochastic process generating the sequence of data and $(X_1, \dots, X_n)$ is the first n inputs in that sequence. The entropy rate of this is defined in equation (17) as

$$h(X) = \lim_{n \rightarrow \infty} \frac{1}{n} H(X_1, \dots, X_n), \quad (17)$$

where $H(X_1, \dots, X_n)$ is the entropy of the first n inputs, and n is the number of steps over which the entropy is computed as it goes to infinity, which measures the effective unpredictability of the input sequence per step. When samples are loaded from a single source in sequence, as in traditional single channel loading, back-to-back samples are highly correlated, which lowers the entropy rate. Alternating between sources using selective embedding reduces this correlation and raises the entropy rate, meaning the model encounters more distinct and informative inputs at each optimization step. This increases information per step which is quantified as the mutual information between the sampled gradient and the population gradient, shown in equation (18).

$$I(\nabla\ell(f_\theta(x_t), y_t), \nabla\mathcal{L}(\theta)) \quad (18)$$

where I is the mutual information between two random variables, $\nabla\ell(f_\theta(x_t), y_t)$ is the per-example gradient of the loss at step t , and $\nabla\mathcal{L}(\theta)$ is the gradient of the loss defined in equation (14). The connection between entropy rate, autocorrelation, and gradient variance is well established in the literature, where a higher entropy rate reduces correlation and lowers the variance of SGD updates [57], [68].

6.5 Benefits of Selective Embedding

6.5.1 Effective Sample Size

When training samples are correlated, the model does not benefit equally from each sample as it would if the samples were independent. The effective sample size, shown in equation (19), captures how much the correlation between consecutive samples reduces the useful information available for generalization.

$$n_{eff} = \frac{n}{1 + 2 \sum_{k=1}^{\infty} \rho(k)} \quad (19)$$

In this case, the denominator increases with correlation, shrinking n_{eff} below the actual number of samples n . By alternating between sources, selective embedding reduces the short lag autocorrelations $\rho(k) = \text{Corr}(\ell_t, \ell_{t+k})$ in the loss sequence, where k is the number of steps between two samples being compared and $\ell_t = \ell(f_\theta(x_t), y_t)$, the per-example loss, consistent with equation (14), evaluated at step t , which increases n_{eff} and improves generalization without requiring additional data [69].

6.5.2 Gradient Variance Reduction

The variance of the batch B gradient depends not only on the individual sample gradients but also on the covariance between them, as shown in equation (20). Here, B is the batch size (i.e., the number of samples in the batch) and g is the batch gradient, defined as the average of the individual sample gradients, $g = \frac{1}{B} \sum_i g_i$.

$$\text{Var}(g) = \frac{1}{B} \sum_i \text{Var}(g_i) + \frac{2}{B} \sum_{i < j} \text{Cov}(g_i, g_j) \quad (20)$$

where g_i and g_j are the per-sample gradient $g(x_i, y_i) = \nabla_{\theta} \ell(f_{\theta}(x_i), y_i)$, consistent with equation (16), evaluated at the i -th and j -th samples in the batch, respectively, and where the sum over $i < j$ runs over all distinct pairs of samples in the batch. When samples in the batch are taken from the same source, their gradients tend to be in the same direction, which will boost the covariance term and, therefore, increase the overall variance of the gradients. Selective embedding changes the composition of each batch by sampling from various sources, which reduces the covariance among per sample gradients $g(x_i, y_i) = \nabla_{\theta} \ell(f_{\theta}(x_i), y_i)$ and lowers the variance of the batch gradient. This means each optimization step provides a more accurate estimate of the true gradient direction leading to more stable and efficient training.

6.5.3 Uniform Mixture Approximation

When training data is drawn from a single source, the empirical distribution seen by the model during training may differ from the true underlying data distribution, which is a mixture of all available sources. Selective embedding enforces per-batch

source balance so that each batch approximates the mixture distribution defined in equation (21).

$$\mathcal{M} = \sum_s \pi_s \mathcal{M}_s, \sum_s \pi_s = 1, [70] \quad (21)$$

Here, s indexes the sources, consistent with the domain and source notation introduced earlier in this chapter, $\mathcal{M}$ is the mixture distribution over (x, y) pairs seen during training, $\mathcal{M}_s$ is the distribution over (x, y) pairs from source s alone, and π_s is the mixture weight, the proportion of training samples drawn from source s , where $\sum_s \pi_s = 1$. This mixture is achieved through the alternation schedule $\sigma(t)$ introduced in section 6.3. Since $\sigma(t)$ alternatively cycles through the available sources, as shown in Algorithm 1 (Table 5-1), each source is selected in turn before repeating. This produces an empirical batch composition in which each source contributes an equal share, directly producing the mixture distribution $\mathcal{M}$ defined in equation (21). By aligning the empirical training distribution with the underlying mixture, selective embedding reduces the mismatch between the training and testing conditions, which is a key driver of cross domain generalization failure in traditional loading methods.

6.5.4 Generalization via Stability

A well known result in PAC-Bayes theory [71] is that the gap between training and test performance depends on both gradient variance and the correlation structure of the training stream. When inputs are highly correlated, the model is more likely to memorize source specific features rather than learning generalizable fault patterns, widening the generalization gap. By decorrelating inputs through source alternation, selective embedding reduces this memorization pressure and tightens the

generalization bound, meaning the gap between training and test performance is expected to be smaller compared to traditional single channel loading.

6.5.5 Risk Decomposition

The overall risk of a model, defined in equation (22), is the same as the expected loss introduced in equation (14), denoted here as $R(\theta)$ to align with standard terminology in statistical learning theory.

$$R(\theta) = \mathcal{L}(\theta) = \mathbb{E}_{(x,y)}[\ell(f_\theta(x), y)] \quad [72] \quad (22)$$

$R(\theta)$ is identical to the training objective defined in equation (14), but renamed here only to match the standard notation used when comparing population risk against the empirical risk $\hat{R}_{SE}(\hat{\theta}_{SE})$ achieved under selective embedding, as required to state the generalization bound.

In practice, this population risk is approximated empirically using the training samples available. The quality of this approximation depends on how those samples are drawn. Selective embedding does not change the underlying data distribution but modifies the sampling process by enforcing per-batch source balance and alternation, which reduces inter-sample dependence and aligns the empirical source proportions with the underlying mixture. This improves both the estimation error, which decreases as $(O(1/\sqrt{n_{eff}}))$, and the shift error, which is reduced through balanced source sampling.

Under the assumption that each source follows a stationary and β -mixing sequence and that the per-example loss ℓ is sub-Gaussian with variance parameter

σ^2 , equation (23) provides a bound on the generalization gap under selective embedding with probability $1 - \delta$.

$$R(\hat{\theta}_{SE}) - \hat{R}_{SE}(\hat{\theta}_{SE}) \leq C_1 \sqrt{\frac{c}{n}} \sqrt{1 + 2 \sum_{k \leq K} \bar{\alpha}(k)} + C_2 \sqrt{\frac{\log(1/\delta)}{n}}, [73] \quad (23)$$

where $\hat{\theta}_{SE}$ is the vector of model parameters learned under selective embedding training, and $\hat{R}_{SE}(\hat{\theta}_{SE})$ is the empirical risk, represented as the training loss achieved by $\hat{\theta}_{SE}$ under selective embedding. δ is the failure probability, so that the bound holds with a probability of at least $1 - \delta$. C_1 and C_2 are positive constants independent of n . Additionally, c is a constant related to the sub-Gaussian variance parameter σ^2 introduced above. K represents the maximum lag included in the summation, and $\bar{\alpha}(k)$ is the β -mixing coefficient at lag k , which quantifies the dependence between samples separated by k steps under the β -mixing assumption stated above. A sequence is said to be β -mixing if the statistical dependence between samples decays as the separation between them increases.

The bound shows that reducing autocorrelation $\rho(k)$ through source alternation raises n_{eff} above the naive single channel case, suggesting tighter generalization behavior consistent with the accuracy improvements reported in Chapter 5. It is important to note that this thesis does not claim a new PAC-Bayes theorem but rather applies existing theoretical tools to provide a principled explanation for those empirical gains, connecting the practical loading method to established results in statistical learning theory.

This bound can be compared to studying for an exam. If a student rereads the same textbook ten times in a row, each repeated reading adds very little new understanding, since the material is redundant with what was just reviewed. If

instead the student studies ten different textbooks on the same topic once each, the student covers different variations of the same material in the same amount of study time, providing them with a deeper understanding and a better preparation for questions they have not seen before. Selective embedding applies this same principle to model training: rather than repeatedly drawing samples from one source, which is similar to rereading the same textbook, it alternates between sources, allowing each training step to introduce distinct variations of the information. Equation (23) formalizes this intuition, showing that this reduction in repeated, correlated information tightens the gap between how well the model performs on training data and how well it is expected to perform on new, unseen data.

6.5.6 Practical Takeaway

To summarize, selective embedding does not enlarge the dataset or modify the model architecture. Instead, it improves the information efficiency of each optimization step by ensuring that the model encounters diverse inputs at every update. By decorrelating inputs through source alternation, selective embedding produces an accurate gradient signal and accelerates convergence, meaning fewer updates are required to reach the same level of generalization compared to traditional single channel loading. This is valuable in cross domain bearing fault diagnosis where data is limited and generalization across unseen operating conditions is the primary challenge. These benefits are achieved without introducing additional learnable parameters, making selective embedding a practical improvement that can be applied to any existing ML architecture and dataset with multiple available sources.

Chapter 7

Conclusion

7.1 Summary of Completed Work

This thesis investigated how data efficiency, rather than dataset size, can drive improvements in machine fault diagnosis using DL. The early chapters established a mechanical understanding of rolling element bearings and the fundamental challenges of fault diagnosis, providing a framework for linking bearing signature theory with data-driven approaches.

Chapter 3 introduces existing bearing datasets, signal processing methods, and ML architectures used throughout this thesis, forming the cross domain framework needed to benchmark input signal length and preprocessing method selection.

In Chapter 4, this framework was applied to determine which input signal length, and preprocessing method provides the highest testing accuracy. The highest testing accuracy for cross domain fault diagnosis achieved was 82.3% using a 1D CNN architecture on the triaxial bearing dataset, attained using an input signal of 1024 and FFT preprocessing. Therefore, for the remainder of the thesis an input signal of 1024 and FFT preprocessing was used. Additionally, a new bearing dataset design

was proposed to overcome the main weaknesses of existing datasets highlighted in Chapter 3, such as limited fault diversity, dependency on seeded faults, and non-distinct bearings during testing. This dataset is designed to be modular, replicable and compatible with traditional and DL workflows.

A novel selective embedding data loading method was presented in Chapter 5, which alternates sensor inputs to improve diversity during training without requiring additional data. Experiments confirmed the relevance of loading methods as a critical design choice, showing that selective embedding greatly enhances cross domain generalization with accuracies up to 90%+.

Theoretical foundations for selective embedding were discussed in Chapter 6. This chapter defined the problem, how source alternation reduces redundancy and increases diversity in the training stream and why these properties lead to improved generalization across domains.

Overall, these chapters demonstrate that in machine fault diagnosis with DL, bigger datasets and increasingly complex models are not the only path to progress. Instead, carefully structured datasets, robust preprocessing, and intelligent data loading methods can allow small but well-curated data to outperform sheer volume. In particular, the selective embedding method shows that when data are loaded in a way that maximizes diversity and representativeness, the choice of algorithm or the size of the dataset becomes less critical. Therefore, this work establishes that data efficiency not dataset size is the true driver of reliable and generalizable machine

condition monitoring, offering a practical and scalable pathway from controlled experiments to real-world industrial deployment.

7.2 Future Work

Whether the findings of this thesis transfer to other bearing sizes have not been empirically tested, and further validation is required before such generalization can be claimed. While the steel slag dataset used in this thesis represents real industrial data collected during steel manufacturing, the remaining datasets originate from laboratory test rigs under controlled conditions. A key direction for future work is expanding validation to additional industrial environments where operating conditions vary continuously in speed, load, and temperature, and where fault progression occurs naturally over time. This would allow the three bearing health conditions of healthy, developing fault, and faulty established in Chapter 4 to be extended to prognostics, specifically the estimation of remaining useful life. A complementary direction is the integration of physics-based digital twin models as synthetic data sources within the selective embedding framework, providing data for fault states that are experimentally difficult to reproduce. Collectively, these extensions would allow for a pilot study quantifying the downtime reduction achievable through early fault detection, providing the economic justification required to translate the data efficiency contributions of this thesis into industrial deployment.

References

- [1] S. Mushtaq, M. M. M. Islam, and M. Sohaib, “Deep Learning Aided Data-Driven Fault Diagnosis of Rotatory Machine: A Comprehensive Review,” *Energies*, vol. 14, p. 5150, Aug. 2021, doi: 10.3390/en14165150.
- [2] S. Nandi, H. A. Toliyat, and X. Li, “Condition Monitoring and Fault Diagnosis of Electrical Motors—A Review,” *IEEE Trans. Energy Convers.*, vol. 20, no. 4, pp. 719–729, Dec. 2005, doi: 10.1109/TEC.2005.847955.
- [3] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, Art. no. 7553, May 2015, doi: 10.1038/nature14539.
- [4] H. Prashad, “Effect of operating parameters on the threshold voltages and impedance response of non-insulated rolling element bearings under the action of electrical currents,” *Wear*, vol. 117, no. 2, pp. 223–240, Jun. 1987, doi: 10.1016/0043-1648(87)90257-2.
- [5] H. Prashad, *Tribology in Electrical Environments*. Elsevier, 2005.
- [6] “Bearing Test Rigs,” NES Bearing Co., Inc. Accessed: Oct. 08, 2021. [Online]. Available: <https://nesbearings.com/bearing-test-rigs/>
- [7] “Bearing Test Rigs - Elgeti Engineering.” Accessed: Oct. 08, 2021. [Online]. Available: <https://www.elgeti-engineering.de/en/products/bearing-test-rigs/>
- [8] R. Jurecki, E. Pokropiński, D. Wieckowski, and Ł. Żołądek, “Design of a Test Rig for the Examination of Mechanical Properties of Rolling Bearings,” *Manag. Syst. Prod. Eng.*, vol. 25, Mar. 2017, doi: 10.1515/mspe-2017-0003.

- [9] M. Sehri, “Physics-informed unit-normalized selective embedding with harmonic loss for fault identification using vibration and acoustic signals,” *Mech. Syst. Signal Process.*, vol. 259, p. 114922, 2026.
- [10] P. R. Anoopnath, V. Suresh Babu, and A. K. Vishwanath, “Hertz Contact Stress of Deep Groove Ball Bearing,” *Mater. Today Proc.*, vol. 5, no. 2, Part 1, pp. 3283–3288, Jan. 2018, doi: 10.1016/j.matpr.2017.11.570.
- [11] R. B. Randall and J. Antoni, “Rolling element bearing diagnostics—A tutorial,” *Mech. Syst. Signal Process.*, vol. 25, no. 2, pp. 485–520, Feb. 2011, doi: 10.1016/j.ymssp.2010.07.017.
- [12] “BALL BEARING BASICS AND TYPES,” ENGINEERING APPLICATIONS. Accessed: Aug. 16, 2023. [Online]. Available: <https://www.hkdivedi.com/2016/02/ball-bearing-basics-and-types.html>
- [13] “Deep groove ball bearings | SKF.” Accessed: Aug. 16, 2023. [Online]. Available: <https://www.skf.com/au/products/rolling-bearings/ball-bearings/deep-groove-ball-bearings>
- [14] “Skf Ball Bearing,” indiamart.com. Accessed: Aug. 16, 2023. [Online]. Available: <https://www.indiamart.com/proddetail/skf-ball-bearing-2850651442855.html>
- [15] W. A. Smith and R. B. Randall, “Rolling element bearing diagnostics using the Case Western Reserve University data: A benchmark study,” *Mech. Syst. Signal Process.*, vol. 64–65, pp. 100–131, Dec. 2015, doi: 10.1016/j.ymssp.2015.04.021.

- [16] “6000 - Deep groove ball bearings | SKF.” Accessed: Jun. 01, 2023. [Online]. Available: <https://www.skf.com/in/products/rolling-bearings/ball-bearings/deep-groove-ball-bearings/productid-6000>
- [17] “708/1250 AMB - Angular contact ball bearings | SKF.” Accessed: Jun. 01, 2023. [Online]. Available: <https://www.skf.com/us/products/rolling-bearings/ball-bearings/angular-contact-ball-bearings/single-row-angular-contact-ball-bearings/productid-708%2F1250%20AMB>
- [18] H. Nagel, “Motor Speeds Explained: Diving into AC and DC Motors | Groschopp Blog,” Groschopp. Accessed: Jun. 01, 2023. [Online]. Available: <https://www.groschopp.com/motor-speeds-explained-ac-dc-motors/>
- [19] “CWRU Bearing Dataset.” Accessed: Dec. 29, 2024. [Online]. Available: <https://www.kaggle.com/datasets/brjapon/cwru-bearing-datasets>
- [20] D. Scott, “Bearing failures diagnosis and investigation,” *Wear*, vol. 25, no. 2, pp. 199–213, Aug. 1973, doi: 10.1016/0043-1648(73)90072-0.
- [21] L. Reuben and D. Mba, “Bearing time-to-failure estimation using spectral analysis features,” *Struct. Health Monit.*, vol. 13, pp. 219–230, Mar. 2014, doi: 10.1177/1475921713520029.
- [22] M. Sehri, P. Dumond, and M. Bouchard, “Design of a Data Fusion and Deep Transfer Learning Test Rig for Roller Bearings Diagnosis and Prognosis,” presented at the ASME 2023 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference,

- American Society of Mechanical Engineers Digital Collection, Nov. 2023. doi: 10.1115/DETC2023-117264.
- [23] F. Haseeb, M. Peng, F. Arshad, K. Awais, and W. Ali, “Hybrid data augmentation for uncertainty mitigation in machine learning based fault diagnosis of pressurized water reactor nuclear power plants,” *Nucl. Eng. Technol.*, vol. 58, no. 11, p. 104498, Nov. 2026, doi: 10.1016/j.net.2026.104498.
- [24] M. Sehri, A. Beatriz Cardoso, F. de Assis Boldt, and P. Dumond, “Industrial steel slag flow data loading method for deep learning applications,” *Expert Syst. Appl.*, vol. 300, p. 130457, Mar. 2026, doi: 10.1016/j.eswa.2025.130457.
- [25] O. Kilim, A. Olar, T. Joó, T. Palicz, P. Pollner, and I. Csabai, “Physical imaging parameter variation drives domain shift,” *Sci. Rep.*, vol. 12, no. 1, p. 21302, Dec. 2022, doi: 10.1038/s41598-022-23990-4.
- [26] M. Sehri, Z. Hua, F. de A. Boldt, and P. Dumond, “Selective embedding for deep learning,” *Knowl.-Based Syst.*, vol. 330, p. 114535, Nov. 2025, doi: 10.1016/j.knosys.2025.114535.
- [27] D. Kumar, S. Mehran, M. Z. Shaikh, M. Hussain, B. S. Chowdhry, and T. Hussain, “Triaxial bearing vibration dataset of induction motor under varying load conditions,” *Data Brief*, vol. 42, p. 108315, Jun. 2022, doi: 10.1016/j.dib.2022.108315.
- [28] T. W. Rauber, A. L. da Silva Loca, F. de A. Boldt, A. L. Rodrigues, and F. M. Varejão, “An experimental methodology to evaluate machine learning methods

- for fault diagnosis based on vibration signals,” *Expert Syst. Appl.*, vol. 167, p. 114022, Apr. 2021, doi: 10.1016/j.eswa.2020.114022.
- [29] J. Hendriks, P. Dumond, and D. A. Knox, “Towards better benchmarking using the CWRU bearing fault dataset,” *Mech. Syst. Signal Process.*, vol. 169, p. 108732, Apr. 2022, doi: 10.1016/j.ymssp.2021.108732.
- [30] “Apparatus & Procedures | Case School of Engineering.” Accessed: Nov. 08, 2024. [Online]. Available: <https://engineering.case.edu/bearingdatacenter/apparatus-and-procedures>
- [31] D. Kumar, S. Mehran, M. Z. Shaikh, M. Hussain, B. S. Chowdhry, and T. Hussain, “Triaxial Bearing Vibration Dataset of Induction Motor under Varying Load Conditions,” vol. 2, Jan. 2022, doi: 10.17632/fm6xzxnf36.2.
- [32] M. Sehri, L. M. Ayres, F. de Assis Boldt, P. Dumond, and M. A. de Souza Leite Cuadros, “An accelerometer-based dataset for monitoring slag in steel manufacturing,” *BMC Res. Notes*, vol. 18, no. 1, p. 420, Oct. 2025, doi: 10.1186/s13104-025-07486-8.
- [33] M. Sehri and P. Dumond, “University of Ottawa Electric Motor Dataset – Vibration and Acoustic Faults under Constant and Variable Speed Conditions (UOEMD-VAFCVS),” vol. 1, Dec. 2023, doi: 10.17632/msxs4vj48g.1.
- [34] M. Sehri and P. Dumond, “University of Ottawa Constant and Variable Speed Electric Motor Vibration and Acoustic Fault Signature Dataset,” *Data Brief*, p. 110144, Feb. 2024, doi: 10.1016/j.dib.2024.110144.

- [35] A. Ding, Y. Qin, B. Wang, L. Guo, L. Jia, and X. Cheng, “Evolvable graph neural network for system-level incremental fault diagnosis of train transmission systems,” *Mech. Syst. Signal Process.*, vol. 210, p. 111175, Mar. 2024, doi: 10.1016/j.ymssp.2024.111175.
- [36] M. Sehri, L. M. Ayres, F. de A. Boldt, P. Dumond, and M. A. de S. L. Cuadros, “Steel Slag Flow Dataset,” vol. 1, Oct. 2024, doi: 10.17632/mjnpsbs6fs.1.
- [37] “Download a Data File | Case School of Engineering.” Accessed: Dec. 09, 2024. [Online]. Available: <https://engineering.case.edu/bearingdatacenter/download-data-file>
- [38] M. Sehri and P. Dumond, “University of Ottawa Rolling-element Dataset – Vibration and Acoustic Faults under Constant Load and Speed conditions (UORED-VAFCLS),” vol. 5, Oct. 2023, doi: 10.17632/y2px5tg92h.5.
- [39] M. Sehri, P. Dumond, and M. Bouchard, “University of Ottawa constant load and speed rolling-element bearing vibration and acoustic fault signature datasets,” *Data Brief*, vol. 49, p. 109327, Aug. 2023, doi: 10.1016/j.dib.2023.109327.
- [40] M. Gupta, R. Wadhvani, and A. Rasool, “A real-time adaptive model for bearing fault classification and remaining useful life estimation using deep neural network,” *Knowl.-Based Syst.*, vol. 259, p. 110070, Jan. 2023, doi: 10.1016/j.knosys.2022.110070.
- [41] B. Li, P. Song, and C. Zhao, “Fusing consensus knowledge: A federated learning method for fault diagnosis via privacy-preserving reference under domain shift,” *Inf. Fusion*, vol. 106, p. 102290, Jun. 2024, doi: 10.1016/j.inffus.2024.102290.

- [42] P. Muhammad Ali and R. Faraj, *Data Normalization and Standardization: A Technical Report*. 2014. doi: 10.13140/RG.2.2.28948.04489.
- [43] C. Lian *et al.*, “CFI-LFENet: Infusing cross-domain fusion image and lightweight feature enhanced network for fault diagnosis,” *Inf. Fusion*, vol. 104, p. 102162, Apr. 2024, doi: 10.1016/j.inffus.2023.102162.
- [44] K. Cabello-Solorzano, I. Ortigosa de Araujo, M. Peña, L. Correia, and A. J. Tallón-Ballesteros, “The Impact of Data Normalization on the Accuracy of Machine Learning Algorithms: A Comparative Analysis,” in *18th International Conference on Soft Computing Models in Industrial and Environmental Applications (SOCO 2023)*, P. García Bringas, H. Pérez García, F. J. Martínez de Pisón, F. Martínez Álvarez, A. Troncoso Lora, Á. Herrero, J. L. Calvo Rolle, H. Quintián, and E. Corchado, Eds., Cham: Springer Nature Switzerland, 2023, pp. 344–353. doi: 10.1007/978-3-031-42536-3_33.
- [45] B. Han *et al.*, “A novel rolling bearing fault diagnosis method based on generalized nonlinear spectral sparsity,” *Measurement*, vol. 198, p. 111131, Jul. 2022, doi: 10.1016/j.measurement.2022.111131.
- [46] B. Samanta and K. R. Al-Balushi, “Artificial Neural Network based fault diagnostics of rolling element bearings using time-domain features,” *Mech. Syst. Signal Process.*, vol. 17, pp. 317–328, Mar. 2003, doi: 10.1006/mssp.2001.1462.
- [47] M. Sehri and P. Dumond, “Exploring Rolling Element Bearing Data Collection and Algorithm Hyperparameters for Machine Learning-Based Fault Diagnosis,”

- Int. J. Progn. Health Manag.*, vol. 16, no. 2, Oct. 2025, doi: 10.36001/ijphm.2025.v16i2.4372.
- [48] S. Hochreiter and J. Schmidhuber, “Long Short-term Memory,” *Neural Comput.*, vol. 9, pp. 1735–80, Dec. 1997, doi: 10.1162/neco.1997.9.8.1735.
- [49] M. Sehri, M. Ertagrin, O. Yildirim, A. Orhan, and P. Dumond, “Deep Learning Approach to Bearing and Induction Motor Fault Diagnosis via Data Fusion,” May 20, 2025, *arXiv*: arXiv:2506.11032. doi: 10.48550/arXiv.2506.11032.
- [50] J. Li, Z. Ye, J. Gao, Z. Meng, K. Tong, and S. Yu, “Fault transfer diagnosis of rolling bearings across different devices via multi-domain information fusion and multi-kernel maximum mean discrepancy,” *Appl. Soft Comput.*, vol. 159, p. 111620, Jul. 2024, doi: 10.1016/j.asoc.2024.111620.
- [51] Z. Li, J. Ma, J. Wu, X. Li, and X. Wang, “Implicit Discriminator Domain Adversarial Residual Network for Cross Domain Rolling Bearing Fault Diagnosis,” *IEEE Trans. Instrum. Meas.*, vol. 73, pp. 1–11, 2024, doi: 10.1109/TIM.2024.3436093.
- [52] M. Mirzaeibonehkhater, M. A. Labbaf-Khaniki, and M. Manthouri, “Transformer-Based Bearing Fault Detection using Temporal Decomposition Attention Mechanism,” Dec. 15, 2024, *arXiv*: arXiv:2412.11245. doi: 10.48550/arXiv.2412.11245.
- [53] Z. Zhao *et al.*, “Applications of Unsupervised Deep Transfer Learning to Intelligent Fault Diagnosis: A Survey and Comparative Study,” *IEEE Trans. Instrum. Meas.*, vol. 70, pp. 1–28, 2021, doi: 10.1109/TIM.2021.3116309.

- [54] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, “Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling,” Dec. 11, 2014, *arXiv*: arXiv:1412.3555. doi: 10.48550/arXiv.1412.3555.
- [55] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778. Accessed: Nov. 12, 2024. [Online]. Available: http://openaccess.thecvf.com/content_cvpr_2016/html/He_Deep_Residual_Learning_CVPR_2016_paper.html
- [56] X. Liu, Y. Hu, and J. Chen, “Hybrid CNN-Transformer model for medical image segmentation with pyramid convolution and multi-layer perceptron,” *Biomed. Signal Process. Control*, vol. 86, p. 105331, Sep. 2023, doi: 10.1016/j.bspc.2023.105331.
- [57] A. Vaswani *et al.*, “Attention is All you Need,” in *Advances in Neural Information Processing Systems*, Curran Associates, Inc., 2017. Accessed: Jun. 26, 2026. [Online]. Available: <https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html>
- [58] M. Sehri *et al.*, “Towards a Universal Vibration Analysis Dataset: A Framework for Transfer Learning in Predictive Maintenance and Structural Health Monitoring,” *Int. J. Progn. Health Manag.*, vol. 16, no. 3, Art. no. 3, May 2025, doi: 10.36001/ijphm.2025.v16i3.4239.

- [59] M. Sehri, N. Khalilian, F. de A. Boldt, M. Bouchard, and P. Dumond, “Integrating Spatial and Temporal Features for Bearing Fault Diagnosis: A Cross-Domain Analysis Using Vibration and Acoustic Data,” *Int. J. Progn. Health Manag.*, vol. 16, no. 2, Oct. 2025, doi: 10.36001/ijphm.2025.v16i2.4469.
- [60] W. Kester, “MT-002 TUTORIAL What the Nyquist Criterion Means to Your Sampled Data System Design,” May 2023.
- [61] J. Guan, L. Wang, C. Zhang, and X. Ma, “Effects of non-metallic inclusions on the crack propagation in bearing steel,” *Tribol. Int.*, vol. 106, pp. 123–131, Feb. 2017, doi: 10.1016/j.triboint.2016.10.030.
- [62] D. Yan, P. Jia, Z. Wang, and L. Zeng, “PMS-DS-RCNet: A Robust Fault Diagnosis Framework for Hydroelectric Generating Units Driven by Random Convolution Augmentation and Parallel Multi-Scale Signal Processing,” *Measurement*, vol. 282, p. 122040, Jul. 2026, doi: 10.1016/j.measurement.2026.122040.
- [63] X. Wang and Y. Lin, “Small Sample Electromagnetic Signal Recognition based on Time Series Data Augmentation,” in *2021 8th International Conference on Dependable Systems and Their Applications (DSA)*, Aug. 2021, pp. 321–326. doi: 10.1109/DSA52907.2021.00049.
- [64] A. Abeysinghe, S. Tohmuang, J. L. Davy, and M. Fard, “Data augmentation on convolutional neural networks to classify mechanical noise,” *Appl. Acoust.*, vol. 203, p. 109209, Feb. 2023, doi: 10.1016/j.apacoust.2023.109209.
- [65] S. Uhlig, I. Alkhasli, F. Schubert, C. Tschöpe, and M. Wolff, “A review of synthetic and augmented training data for machine learning in ultrasonic non-

- destructive evaluation,” *Ultrasonics*, vol. 134, p. 107041, Sep. 2023, doi: 10.1016/j.ultras.2023.107041.
- [66] M. Sehri, T. Yan, G. Vashishtha, and S. Chauhan, “Influence of data loading strategies on multi sensor vibration gearbox fault diagnosis under limited data conditions,” *Int. J. Adv. Manuf. Technol.*, May 2026, doi: 10.1007/s00170-026-18342-9.
- [67] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016. Accessed: Dec. 11, 2023. [Online]. Available: <https://books.google.com/books?hl=en&lr=&id=omivDQAAQBAJ&oi=fnd&pg=PR5&dq=info:MHq4MMenr-gJ:scholar.google.com&ots=MOM0dn1BWS&sig=RTGF5dTNQG3pNt9kaG6v9N6KtK0>
- [68] A. Xu and M. Raginsky, “Information-theoretic analysis of generalization capability of learning algorithms,” in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, in NIPS’17. Red Hook, NY, USA: Curran Associates Inc., Dec. 2017, pp. 2521–2530. Accessed: Mar. 23, 2026. [Online]. Available: <https://dl.acm.org/doi/10.5555/3294996.3295013>
- [69] B. Yu, “Rates of Convergence for Empirical Processes of Stationary Mixing Sequences,” *Ann. Probab.*, vol. 22, no. 1, pp. 94–116, Jan. 1994, doi: 10.1214/aop/1176988849.
- [70] Y. Mansour, M. Mohri, and A. Rostamizadeh, “Domain Adaptation with Multiple Sources,” in *Advances in Neural Information Processing Systems*, Curran

- Associates, Inc., 2008. Accessed: Sep. 25, 2026. [Online]. Available: <https://proceedings.neurips.cc/paper/2008/hash/0e65972dce68dad4d52d063967f0a705-Abstract.html>
- [71] D. A. McAllester, “PAC-Bayesian model averaging,” in *Proceedings of the twelfth annual conference on Computational learning theory*, in COLT ’99. New York, NY, USA: Association for Computing Machinery, Jul. 1999, pp. 164–170. doi: 10.1145/307400.307435.
- [72] L. Ralaivola, M. Szafranski, and G. Stempfel, “Chromatic PAC-Bayes bounds for non-iid data: Applications to ranking and stationary β -mixing processes,” *J. Mach. Learn. Res.*, vol. 11, pp. 1927–1956, 2010.
- [73] M. Mohri and A. Rostamizadeh, “Rademacher Complexity Bounds for Non-I.I.D. Processes,” in *Advances in Neural Information Processing Systems*, Curran Associates, Inc., 2008. Accessed: Sep. 25, 2026. [Online]. Available: <https://proceedings.neurips.cc/paper/2008/hash/7eacb532570ff6858afd2723755ff790-Abstract.html>