FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO




# Sensor System for Predictive Maintenance in Industrial Environment

**André Cunha Enes Gonçalves**

Mestrado em Engenharia Eletrotécnica e de Computadores

Supervisor: Vítor Hugo Machado Oliveira Pinto

October 17, 2023

© André Cunha Enes Gonçalves, 2023

# Resumo

Com a chegada da Indústria 4.0 e do aumento da acessibilidade de *hardware* capaz de capturar e processar grandes quantidades de dados complexos, a manutenção de equipamentos industriais tem passado por vários avanços tecnológicos, mais especificamente, no que concerne à avaliação da condição e *performance* dos equipamentos. Estes avanços possibilitaram o surgimento da Manutenção Preditiva, que usa dados em tempo real para prever futuras falhas iminentes em equipamentos, minimizando tempo de inatividade dos equipamentos e reduzindo custos de manutenção.

A maioria das operações de manufatura dependem de equipaments tecnologicamente ultrapassados, ditos *legacy*, que não possuem os benefícios de sensores integrados que possibilitam a incorporação de Manutenção Preditiva nas linhas de produção.

Avanços na *Internet of Things* e no ramo de *Machine Learning* motivaram o tema e objetivo desta dissertação: a projeção de um sistema sensorial capaz de integrar Manutenção Preditiva em sistemas *legacy*, mais especificamente num *Automated Guided Vehicle* e nas máquinas envolventes.

O sistema tirará proveito de sensores de vibração, som e temperatura para obter os dados e de algoritmos de *Machine Learning* para a deteção de anomalias. Uma análise aprofundada do tipo de sensor e das características dos mais usados na literatura para medir as grandezas mencionadas foi realizada, tendo em conta o custo do sistema completo e as limitações físicas, em termos de espaço, do veículo.

Para testar o conceito e comprovar a metodologia utilizada, o sistema foi utilizado em plataformas de testes, simulando o *Automated Guided Vehicle*, para detetar comportamento anómalo. O pré-processamento utilizado baseia-se em características temporais para reduzir a quantidade de dados enviados à distância. A técnica deu origem a resultados praticamente ideais para os modelos de *Supervised Learning*, enquanto os resultados dos modelos de *Unsupervised Learning* revelaram-se relativamente fracos. Adicionalmente, uma comparação entre sensores de preços diferentes foi efetuada.

# Abstract

In line with the Industry 4.0 Revolution, equipment maintenance is undergoing several advancements which take advantage of the ever-increasing affordability of hardware capable of capturing and processing high amounts of complex data to analyse equipment health and performance. This progress has given rise to Predictive Maintenance, a proactive strategy that uses real-time data analysis to predict equipment failures, which optimises schedules, minimises downtime and reduces maintenance costs by replacing components only when necessary.

A substantial portion of manufacturing operations depend heavily on legacy equipment to meet their production demands, which lack the advantages of contemporary sensor technology to incorporate Predictive Maintenance in the production line.

By leveraging technology developed for the Internet of Things and state-of-the-art Machine Learning algorithms, this dissertation's ultimate goal is to design a wireless sensor system capable of bringing Predictive Maintenance to legacy machinery, more specifically to an Automated Guided Vehicle and its surrounding machines.

The system aims to achieve its objective through the use of vibration, sound and temperature sensors as well as with Machine Learning algorithms. A thorough analysis of the sensor technology and characteristics employed in the literature of these physical quantities was made, keeping in mind the physical space constraints of the vehicle and the ultimate cost of the sensor system.

The system concept was put to the test by detecting anomalous behaviour in testing platforms meant to replicate an Automated Guided Vehicle. The preprocessing method used is based on time-domain features to minimise the amount of data to send wirelessly. The technique yielded nearly ideal predictions for the selected Supervised Learning models, while the performance of the Unsupervised Learning models was notably poor. Additionally, a comparison between sensors, which varied in cost, was conducted.

# Acknowledgements

It took me some time to realise I had fallen into a black hole of possibilities. Without the help of many people, I am not sure I would be able to climb out.

First, I would like to thank my supervisor, Professor Vítor Pinto, for giving me the opportunity to be the first student to take on a new project take will be carried out by other students in the near future. I would still be worrying about insignificant details without your guidance. I would also like to thank DIGI2 Laboratory for providing all the extra equipment that I would not have access to if I was to do my dissertation elsewhere.

My family is responsible for the dedication I put into this project. If not for them, I would not know how to keep working, even though, sometimes, motivation vanishes. It came back.

I would have gone insane if not for the help and care of all my friends. Why do anything if you can't have some fun doing it?

Thank you

André Enes

*"Quem não tem cão,*
*caça com gato."*

Ditado Popular

# Contents


**1 Introduction** **1**
- 1.1 Context . . . 1
- 1.2 Motivation . . . 1
- 1.3 Objective . . . 2
- 1.4 Document Structure . . . 3

**2 Background** **5**
- 2.1 Nomenclature Disambiguation . . . 5
- 2.2 Types of Maintenance . . . 6
  - 2.2.1 Reactive Maintenance . . . 6
  - 2.2.2 Preventive Maintenance . . . 7
  - 2.2.3 Predictive Maintenance . . . 7
- 2.3 Signal Processing . . . 8
  - 2.3.1 Envelope . . . 8
  - 2.3.2 Discrete Fourier Transform . . . 8
  - 2.3.3 Signal Filtering . . . 9
  - 2.3.4 Short-Time Fourier Transform . . . 10
  - 2.3.5 Wavelet Transform . . . 11
  - 2.3.6 Empirical Wavelet Transform . . . 12
  - 2.3.7 Empirical Mode Decomposition . . . 13
  - 2.3.8 Hilbert Transform . . . 14
  - 2.3.9 Hilbert-Huang Transform . . . 14

**3 Related Work** **15**
- 3.1 Critical Components . . . 15
- 3.2 Data Collection . . . 17
  - 3.2.1 Vibration Analysis . . . 17
  - 3.2.2 Acoustic Analysis . . . 18
  - 3.2.3 Acoustic Emission Analysis . . . 19
  - 3.2.4 Temperature Analysis . . . 20
  - 3.2.5 Oil Analysis . . . 20
  - 3.2.6 Electric Current Analysis . . . 21
- 3.3 Prediction Algorithms . . . 21
  - 3.3.1 Prediction Problem Definition . . . 22
  - 3.3.2 Statistical Methods . . . 23
  - 3.3.3 Supervised Learning Methods . . . 24
  - 3.3.4 Semi-Supervised and Unsupervised Learning Methods . . . 26
- 3.4 Preprocessing . . . 28

3.4.1 Outlier Removal . . . 29
3.5 Digital Signal Processing . . . 30
3.5.1 Discrete Fourier Transform . . . 31
3.5.2 Short-Time Fourier Transform . . . 31
3.5.3 Time-frequency Techniques . . . 32
3.6 Wireless Connection . . . 32
3.7 Testing Platforms . . . 33
3.8 Summary . . . 34

**4 Methodology** **37**
4.1 Prototype Design . . . 37
4.1.1 Vibration Sensors . . . 37
4.1.2 Microphones . . . 39
4.1.3 Temperature Sensors . . . 46
4.1.4 Microcontroller . . . 47
4.1.5 Voltage Incompatibility . . . 48
4.1.6 Analog-to-Digital Conversion . . . 49
4.1.7 Hardware Connections . . . 49
4.1.8 Local Processing . . . 49
4.1.9 WiFi Connection . . . 50
4.1.10 Overview . . . 53
4.2 Database . . . 53
4.3 Custom Testbed . . . 54
4.3.1 Tested Components . . . 57
4.4 Software Structure . . . 58
4.4.1 Data Collection . . . 58
4.4.2 Dataset Creation . . . 62
4.4.3 ML Models . . . 64

**5 Experimental Results** **67**
5.1 Metrics . . . 67
5.2 Test Structure . . . 69
5.2.1 Dataset Unbalance . . . 69
5.2.2 Sensor Comparison . . . 70
5.2.3 ML Automation Script . . . 70
5.3 Ultrasonic Microphone Measurements . . . 70
5.4 Temperature Analysis . . . 72
5.5 Unsupervised Models Results . . . 74
5.6 Supervised Models Results . . . 75
5.6.1 Sensor Comparison . . . 76
5.6.2 Unknown Anomalies . . . 77
5.7 Summary . . . 78

**6 Conclusion and Future Work** **81**
6.1 Conclusion . . . 81
6.2 Future Work . . . 82

**References** **85**

# List of Figures


2.1 Nomenclature Venn Diagram . . . 7
2.2 Envelope . . . 9
2.3 Example STFT . . . 11
2.4 Ricker and Morlet Wavelets . . . 12
2.5 Example Wavelet Transform . . . 12
2.6 Example Empirical Mode Decomposition . . . 13

3.1 Critical Components Diagram . . . 16
3.2 IRT Image of an AC Motor . . . 21
3.3 Ball Bearing Internal Structure . . . 24
3.4 One-Class SVM Example . . . 27
3.5 Autoencoder Structure . . . 28
3.6 PRONOSTIA testbed . . . 34
3.7 SpectraQuest Simulator . . . 35

4.1 *ADXL355* Accelerometer Evaluation Board . . . 38
4.2 *MMA8451* Accelerometer Breakout Board . . . 39
4.3 GRAS 40PH Frequency Response . . . 41
4.4 Digital Microphone from *Adafruit* . . . 42
4.5 Digital Microphone Frequency Response . . . 42
4.6 Ultrasonic Microphone Breakout Board . . . 43
4.7 Ultrasonic Microphone Frequency Response . . . 44
4.8 Microphone Amplifier Circuit . . . 46
4.9 *MLX90514* Infrared Thermometer . . . 47
4.10 Teensy 4.1 . . . 48
4.11 Logic Level Converter . . . 48
4.12 Module Connection Diagram . . . 50
4.13 ESP-12E . . . 51
4.14 FT232RL . . . 52
4.15 ESP-12E Programming Mode . . . 52
4.16 ESP-12E Serial . . . 52
4.17 Module Overview Diagram . . . 53
4.18 Testbed 1 in a bare state . . . 56
4.19 Testbed 2 in a bare state . . . 56
4.20 Testbed 3 . . . 56
4.21 Ball Bearing in normal state . . . 57
4.22 Ball Bearing in anomalous state . . . 58
4.23 Main Finite State Machine - SD Card . . . 59

4.24 Continuous Recording Finite State Machine . . . . . 59
4.25 Read sensor data Algorithm . . . . . 60
4.26 Save data in SD card Algorithm . . . . . 61
4.27 Dataset Structure . . . . . 64

5.1 Confusion Matrix Example . . . . . 68
5.2 ML Automation Script Flowchart . . . . . 71
5.3 Ultrasonic Microphone Readings . . . . . 71
5.4 Zoomed In Ultrasonic Microphone Readings . . . . . 72
5.5 Temperature Readings - Testbed 2 . . . . . 73
5.6 Zoomed In Temperature Readings . . . . . 73
5.7 Isolation Forest Confusion Matrix . . . . . 75
5.8 XGBoost Confusion Matrix . . . . . 76
5.9 Confusion Matrix of model in Table 5.7 . . . . . 79

# List of Tables


3.1 IoT communication protocol comparison . . . . . 33

4.1 Accelerometer Feature Comparison . . . . . 38
4.2 Microphone Feature Comparison . . . . . 41
4.3 SPH0645LM4H's Main Features . . . . . 42
4.4 SPU0410LR5H-QB's Main Features . . . . . 44
4.5 Amp Passive Component Values . . . . . 46
4.6 MLX90614 Specifications . . . . . 47
4.7 Communication Protocol Comparison . . . . . 51

5.1 Results Temperature vs. No Temperature . . . . . 74
5.2 Unsupervised Models Metrics . . . . . 75
5.3 Best Results from Supervised Models . . . . . 76
5.4 More results for the same conditions . . . . . 77
5.5 Sensor Impact on model performance comparison . . . . . 78
5.6 Model metrics combining sensors . . . . . 79
5.7 Best results testing with anomalies not present in the dataset . . . . . 79

# Acronyms and Abbreviations

| | |
|---|---|
| ADC | Analog-to-Digital Converter |
| ANN | Artificial Neural Network |
| CNN | Convolutional Neural Network |
| DMA | Direct Memory Access |
| DSP | Digital Signal Processing |
| EMD | Empirical Mode Decomposition |
| HHT | Hilbert-Huang Transform |
| HSA | Hilbert Spectral Analysis |
| HTTP | Hypertext Transfer Protocol |
| I2C | Inter-Integrated Circuit |
| I2S | Inter-IC Sound |
| IMF | Intrinsic Mode Function |
| IMF | Intrinsic Mode Function |
| IR | Infrared Light |
| IoT | Internet of Things |
| LOF | Local Outlier Factor |
| MEMS | Microelectromechanical Systems |
| ML | Machine Learning |
| MQTT | Message Queuing Telemetry Transport |
| OSI | Open Systems Interconnection |
| PCB | Printed Circuit Board |
| PM | Preventive Maintenance |
| PdM | Predictive Maintenance |
| RF | Random Forest |
| RM | Reactive Maintenance |
| RTD | Resistance Temperature Detector |
| SPI | Serial Peripheral Interface |
| STFT | Short-Time Fourier Transform |
| SVM | Support Vector Machine |
| TCP | Transmission Control Protocol |
| UART | Universal Asynchronous Receiver/Transmitter |
| UDP | User Datagram Protocol |
| USB | Universal Serial Bus |
| k-NN | k-Nearest Neighbours |

# Chapter 1

# Introduction

## 1.1 Context

In an ever-increasing world of rising industrial productivity, with the advent of Industry 4.0, where every process is optimised to the fullest, the maintenance of the machines in charge of this surge cannot be left behind. Traditional maintenance practices often result in excessive downtime, increased repair costs, and inefficient resource allocation. Thankfully, this problem has not gone unnoticed. A new paradigm in industrial maintenance, leveraging state-of-the-art technology, is emerging in modern industry: Predictive Maintenance [1]. This new model uses data collected from machines to estimate when a certain machine part will require either repair or replacement. Older maintenance systems would either correct the anomaly after it occurred or preventively replace parts, without confirming their actual necessity.

A significant challenge in implementing Predictive Maintenance (PdM) lies in integrating modern technologies into older machinery without resorting to replacing the entire equipment with newer, technologically advanced models. Retrofitting these machines with suitable sensors, data acquisition systems, and communication interfaces necessitates careful engineering to ensure seamless compatibility and efficient data integration [2]. A module containing several types of sensors and capable of processing data autonomously which could be used in different machines without significant changes would enable legacy systems to benefit from the advantages of data-driven maintenance techniques while reducing the cost of individually retrofitting older machines.

Recent research developments in the area of Machine Learning (ML) have enabled complex processes being able to be predicted with a high degree of certainty. These advancements also include the field of maintenance, with several different methods being explored and evaluated with favourable results over more traditional methods.

## 1.2 Motivation

Estimates show that maintenance costs account for 15 to 60 % of the total expenses of manufacturing operations [3]. While these costs vary substantially between industries, depending on

how much that production relies on complex and highly specialised equipment, such as the aviation industry, due to the need for specialised technicians, spare parts, and intricate maintenance procedures, whereas a sector like agriculture may have lesser costs in maintenance on account of relatively simpler machinery. Nevertheless, all industries can benefit from the implementation of Predictive Maintenance since it can heavily lower unplanned downtime and lower product defects due to earlier warnings of necessary repairs [4].

It is also important to highlight the impact of defective equipment on the well-being of factory workers. Prolonged exposure to such noisy conditions can give rise to various health conditions, with hearing damage or loss being one of the most glaring concerns. Furthermore, there are more subtle repercussions, involving a decrease in worker productivity due to annoyance, frustration, dissatisfaction and increased stress; the risk of accidents also increases [5]. Although most noise would not be entirely eliminated from a prototypical factory, the deployment of Predictive Maintenance in such an environment would alleviate some of the harmful noise.

Most current implementations of PdM in practical applications make use of existing theoretical models in order to monitor and predict faults in the system under evaluation. Nonetheless, these models require a considerable amount of knowledge about the system and their predictions can be hindered by high amounts of noise which limit the potential use cases. Employing the use of ML techniques facilitates the implementation of Predictive Maintenance in more intricate systems, where there is less knowledge available or in more dynamic conditions where the setting is constantly affected by noise. Machine Learning methods are capable of extracting features from a set of data while being agnostic to the problem at hand. This versatility allows for detecting anomalies in complex data [6].

The domain of the Internet of Things has propelled forward the development of microcontrollers, sensor technology, and lightweight communication protocols, unveiling the potential of energy-efficient devices. The recent interest, in conjunction with the decrease in the cost of electronics that can be used in a Predictive Maintenance context, such as accelerometers and temperature sensors, makes possible the creation of a module capable of collecting relevant data from machines, running algorithms that detect faults and sending meaningful information wirelessly to alert the maintenance department.

## 1.3 Objective

The objective of this dissertation is to develop a system capable of determining when maintenance is required on an Autonomous Mobile Robot and its surrounding machines in an industrial scenario. To reach this objective, a sensor module for PdM was designed and tested. The full system would include software outside the microcontroller to fully process the gathered data to detect faults. The complete system should be able to fulfil the following criteria:

- Collect data from multiple sensors
- Preprocess the data

- Send the data to a database
- Detect anomalous behaviour present in the data

Considering that the final objective is to achieve the capability to detect anomalies in an Automated Guided Vehicle whose assignment is to tug cargo around a factory floor, and since intentionally damaging the vehicle is not possible, a testing platform must be built to evaluate the system and to ensure the procedures function as intended. Different Machine Learning algorithms will also be tested to determine the most effective ones for the given scenario. Although not mandatory, the overall cost of the module should be taken into account by ensuring cost-effectiveness.

## 1.4 Document Structure

This dissertation is composed of six chapters, where the entire process of developing the module is laid out, from researching relevant technologies to the final tests and conclusions. In Chapter 2, some theoretical concepts are discussed, which will be pertinent for the rest of the dissertation. Chapter 3 reviews articles containing work related to the objectives, which components are the most critical for machines, the types of measurements used in the field, signal processing techniques, machine learning algorithms, preprocessing methods and wireless technologies. In Chapter 4, the methodology for developing the module is detailed, analysing how the components were chosen, how the software to run the system works and the method behind the testing procedures. Chapter 5 discusses the outcomes while explaining the metrics utilised and exploring the results. Chapter 6 brings forth the conclusion and emphasises further future improvements.

# Chapter 2

# Background

In this chapter, relevant concepts to the dissertation's objective will be reviewed. Ranging from the existing types of maintenance to signal-processing theoretical base concepts. Additionally, a dedicated section will address and clarify certain nomenclature used in pertinent articles where a definitive standard or consensus is lacking regarding specific definitions.

## 2.1 Nomenclature Disambiguation

Condition Monitoring refers to the continuous assessment and tracking of various parameters or indicators to evaluate the health and performance of a system or equipment [7]. It aims to detect any deviations from normal operating conditions and provides early warnings of potential failures or abnormalities. It is not generally used in other contexts apart from Predictive Maintenance.

The Remaining Useful Life of an asset or system is the duration remaining from the present moment until the conclusion of its productive operational span [8]. Predicting the Remaining Useful Life involves analysing historical data, condition monitoring data, and possibly other factors to estimate when a system might fail or become inefficient. Other definitions are used in other research fields, but the previous one remains the predominant choice when considering system health monitoring.

Fault Detection is defined as the determination of the occurrence of faults in the monitored system. It entails the detection of faults in the processes, actuators and sensors by using dependencies between different measurable signals [9]. A fault is defined as an unpermitted deviation of at least one feature of the system from the standard and acceptable condition [10]. A failure is a permanent interruption of a system's ability to perform a required function under a set of specific operating conditions [11]. The difference between these definitions clarifies that articles referring to Fault Detection tend to focus on a specific part of a bigger machine or structure, as the case with [12], which is a review on bearing fault detection, [13] which is a review on induction motor fault detection and [14] which is about fault detection in vibrating structures. The term Failure Detection is rarely used and normally appears in articles in the Distributed Computing field [15].

Fault Diagnostics delves into the identification and understanding of the root cause behind a detected fault or anomaly. It involves a comprehensive analysis of available data to pinpoint the exact source of the problem. Fault diagnostics aim to answer questions like "What is causing the fault?" and are crucial for effective troubleshooting and targeted repair strategies. It is a more in-depth step following fault detection.

Fault Prognostics take a step beyond fault detection and diagnostic processes. Prognostics involve predicting the future behaviour of a system based on its current state and historical data. This includes estimating the time remaining before a system experiences a specific failure, considering factors like wear and tear, usage patterns, and environmental conditions.

Anomaly Detection is the process of identifying deviations or outliers from normal patterns within a given dataset or system behaviour [16]. Techniques referring to Anomaly Detection can be applied in several different fields, including fraud detection [17], cyber-intrusion detection [18] and image processing [19]. The most relevant articles on Anomaly Detection for this dissertation are those whose focus is on industrial damage detection, which will be further explored. The term is usually meant for moving parts analysis, as with bearings and gearboxes, and not referring to structural or static parts analysis

Figure 2.1 shows how the research terms relate to each other. It is important to note that the circles' proportions do not represent the frequency of keyword occurrence; rather, they serve as a visual aid to enhance comprehension. Condition Monitoring, Anomaly Detection and Fault Detection were the target terms for the research part of this dissertation since the project focus area is on detecting faults in the system under analysis. The scope of this project, as outlined in Figure 2.1, does not encompass tasks such as predicting the Remaining Useful Life of specific components or engaging in fault diagnosis and prognostics. Although the previous keywords were not the priority during the research phase, some techniques that are often employed in these research fields were analysed due to the comparable goals of the articles and of this dissertation.

Another ambiguity found while researching for this dissertation is that some articles refer to Anomaly Detection as Abnormality Detection; however, both terms can be used interchangeably when referring to the same goal: to identify deviations from normal patterns, as previously defined. In PdM, Anomaly Detection is more common, with Abnormality being more common in the medical field, [20] [21].

## 2.2 Types of Maintenance

The impact of maintenance represents a total of **15 to 60%** of the total costs of operating of all manufacturing [22]. As such, it is essential to compare the existing maintenance plan strategies to select the one which minimises these costs.

### 2.2.1 Reactive Maintenance

Reactive Maintenance (RM) (or Corrective Maintenance) is the simplest maintenance strategy as it only happens when a piece of equipment stops working and consists of stopping production to

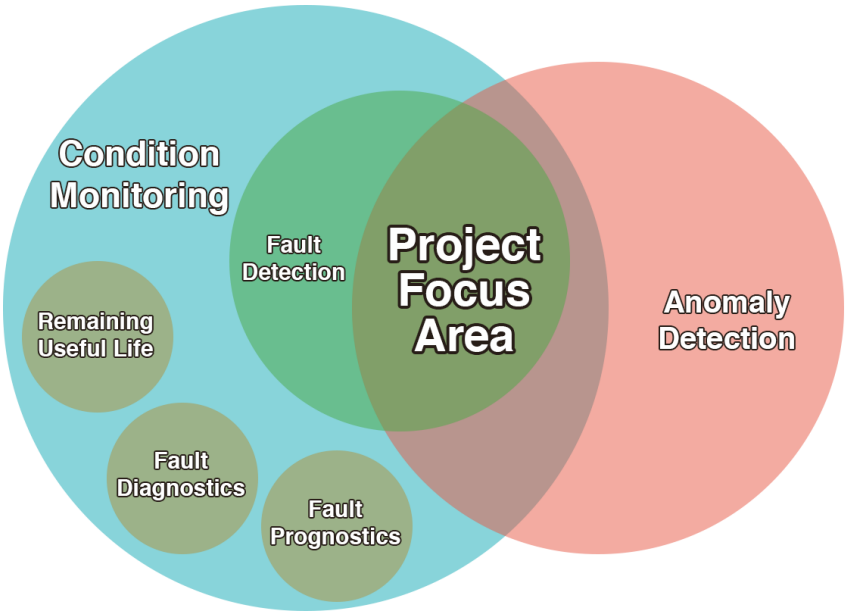



Figure 2.1: Venn Diagram of research terms

repair the failure [23]. RM can lead to increased downtime, production delays, and higher repair costs. Moreover, the unpredictability of breakdowns can impact operational efficiency and hinder long-term asset reliability. As a result, many industries seek to transition from reactive to more proactive maintenance approaches to optimise equipment performance and minimise unplanned disruptions.

### 2.2.2 Preventive Maintenance

Preventive Maintenance (PM) is a technique involving periodic maintenance as to prevent sudden failure. This approach relies on scheduled inspections and repairs to ensure that equipment is kept in good working condition and that any potential problems are identified and addressed before they lead to major breakdowns or downtime, thus saving maintenance costs [23]. It is more effective in reducing costs than RM but repairs and machine parts are replaced regardless of the deterioration state of the piece of equipment, leading to unnecessary corrective actions.

### 2.2.3 Predictive Maintenance

PdM as a concept has long existed, but only recently have emerging technologies become seemingly capable and inexpensive enough to make PdM widely accessible [24]. It has been adopted by many sectors, especially by the ones where reliability is paramount, such as power plants, utilities, transportation systems, communication systems, and emergency services [25]. In essence, it uses data analysis to predict when equipment failure is likely to occur, allowing maintenance to be performed proactively and optimising the evaluation of the monitored system's deterioration state [23].

As mentioned above, PdM uses data to predict future failures by leveraging data analytics, sensor technology, and algorithms to predict and prevent equipment failures before they occur. Unlike traditional maintenance approaches, RM and PM, which rely on fixed schedules or reactive responses to breakdowns, PdM uses real-time data from sensors and historical performance information to forecast potential issues and identify early warning signs of equipment deterioration. By continuously monitoring the health and performance of machinery, PdM enables proactive interventions, such as timely repairs or component replacements, extending the lifespan of equipment and minimising costly downtime and unplanned disruptions to operations. This approach has gained significant traction across industries, including manufacturing, energy, transportation, and healthcare, as it offers increased operational efficiency, reduced maintenance costs, and improved safety, ultimately leading to enhanced productivity and customer satisfaction [24].

## 2.3 Signal Processing

Signal processing is the field that deals with analysing and manipulating signals to extract useful information. It involves techniques like filtering, compression, and analysis to enhance signals and remove noise. It has applications in telecommunications, audio and video processing, medical imaging, and more. Signal processing plays a pivotal role in condition monitoring by extracting meaningful insights from diverse signals like vibration, temperature, and pressure. Through techniques like feature extraction, pattern recognition, noise filtering, and time-frequency analysis, signal processing enables the identification of anomalies, patterns, and changes in signals, which are indicative of system health and potential faults. It aids in enhancing data quality, detecting deviations from normal behaviour, and enabling accurate predictions, thus facilitating informed decision-making for maintenance and operational optimisation.

### 2.3.1 Envelope

In the domain of signal processing, an *envelope* represents a curve that traces the changes in amplitude or intensity of a signal over time. Figure 2.2 shows a signal and its envelope, generated using the *SciPy* [1] Python package.

Envelopes play a role in time-frequency analysis, offering insights into how a signal's frequency content evolves over time. This is particularly valuable in audio processing, enabling the identification of time-varying features in audio signals. Additionally, envelopes serve as a visualisation tool, providing a concise representation of a signal's overall behaviour. They simplify waveform analysis, facilitating tasks like noise reduction and feature extraction [26].

### 2.3.2 Discrete Fourier Transform

The Fourier Transform converts a function into a form that describes the frequencies present in the original function [26]. It is a fundamental technique that is used in many different fields, ranging

[1] https://scipy.org/

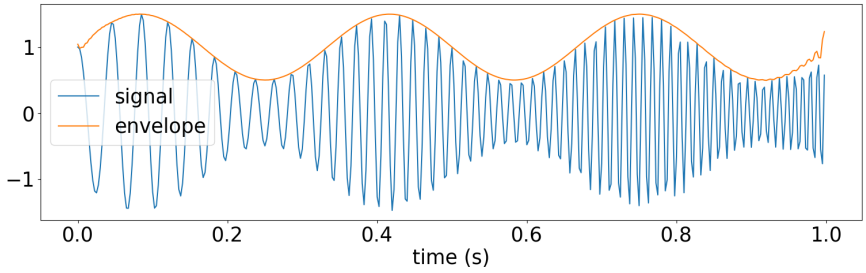



Figure 2.2: Envelope of a signal generated through *SciPy*

from mathematics and physics to image analysis and audio compression. This project's most relevant use case is its use in Signal Processing, which deals with the analysis, manipulation, and interpretation of signals. Signals are defined as a function that conveys information, such as audio, video, images, sensor data, or any other form of measurable data. Signal processing techniques aim to extract useful information from signals, enhance their quality, remove noise, compress them for efficient storage or transmission, and perform various other operations [27]. Signal Processing can be subdivided into two sub-fields: Analog Signal Processing and Digital Signal Processing (DSP); the latter handles signals that need to be analysed using digital hardware, as the case with this dissertation.

The Fourier Transform in the DSP world must be adapted for discrete signals since analog signals cannot be sampled at an infinite rate. Thankfully, the Discrete Fourier Transform (DFT) exists and is widely used in DSP. It enables the analysis of discrete-time signals in the frequency spectrum, and the equation that describes it for a periodic signal is displayed in Equation 2.1 [26].

$$X(k) = \frac{1}{N} \sum_{k=0}^{N-1} x(n) e^{j\left(\frac{2\pi}{N}\right)kn} \tag{2.1}$$

The most challenging obstacle of using this equation lies on the fact that its computation is of $O(N^2)$ complexity, using the *big O notation* [28]. The most well-known and used algorithm to compute the DFT faster and more efficiently is the Fast Fourier Transform (FFT) [29]. The main advantage of using the FFT algorithm is that its complexity is $O(Nlog(N))$, and therefore, it is significantly faster for larger sequences. This speed advantage is crucial for real-time signal processing applications, such as audio and video processing, communication systems, image analysis, and many other domains where processing large amounts of data is required.

### 2.3.3 Signal Filtering

In the realm of Signal Processing, noise poses a substantial challenge, as it can impact the desired outcome at every stage, ranging from the amplification of electrical signals to the subsequent sampling of these signals. Most electrical noise manifests itself as unwanted frequencies in the signal. These undesirable frequencies can be entirely or partly removed from the signal using

analog and/or digital filters. The greatest difference between analog and digital filters lies in the fact that analog filters deal with continuous signals and digital filters with discrete signals. Another divergence is that analog filters are deployed using physical components, e.g. capacitors and inductors, while digital filters are mostly set up using programs processed using a microprocessor after the signal has been sampled with an Analog-to-Digital Converter (ADC) [26]. Different types of filters, which vary in complexity, affect the original signal differently in its frequency response and phase delay, and some of the most common designs will be explored further along.

Analog and digital filters are essential signal processing components used to modify or extract specific frequency components from signals. Analog filters are implemented using analog electronic components such as resistors, capacitors, and inductors. They operate on continuous-time signals and are typically found in analog circuits, audio systems, and radio frequency applications. Analog filters are characterised by their frequency response, which determines how the filter attenuates or amplifies different frequency components in the signal. The design of analog filters involves selecting the appropriate filter type (e.g., low-pass, high-pass, band-pass) and adjusting parameters to achieve the desired frequency response [26].

On the other hand, digital filters process discrete-time signals represented in digital form, such as those found in digital audio processing, image processing, and communication systems. Digital filters are implemented using digital signal processing (DSP) algorithms and are often realised using software or specialised hardware. The design of digital filters is based on mathematical algorithms that manipulate the signal's discrete samples. Digital filters can offer greater flexibility and precision compared to analog filters due to the inherent advantages of digital processing, such as easy reconfigurability, precise control over filter characteristics, and the ability to implement complex filtering operations. Moreover, digital filters can be easily replicated, stored, and transmitted, making them ideal for modern digital systems where data is processed and transmitted in digital format. Both analog and digital filters play critical roles in various engineering applications, ensuring efficient signal processing and signal conditioning in a wide range of systems [26].

### 2.3.4 Short-Time Fourier Transform

The Short-Time Fourier Transform (STFT) is an algorithm meant to increase clarity in the time domain when analysing signals in the frequency spectrum. As previously seen in Subsection 2.3.2, the Fourier Transform enables the transformation of a signal from the time domain to the frequency domain, yet all information about how certain frequencies change throughout time is lost. The STFT tries to solve this issue by dividing the signal into smaller parts and then computing the FFT for every section. This way, it is possible to analyse how the frequencies of the signal change over time [26]. Figure 2.3, taken from [5], displays a signal with environmental noise and a significant event at the 4-second mark. The DFT of this signal would not be able to pinpoint the time at which the increase of amplitude of the signal occurred. By using the STFT, information about time can be extracted from the signal, as well as the different frequencies and their amplitudes, which in

this case, is able to detect an increase in amplitude at 1 kHz at the 4-second mark, which is shown in yellow, meaning a stronger amplitude, according to the scale on the right of the graph.

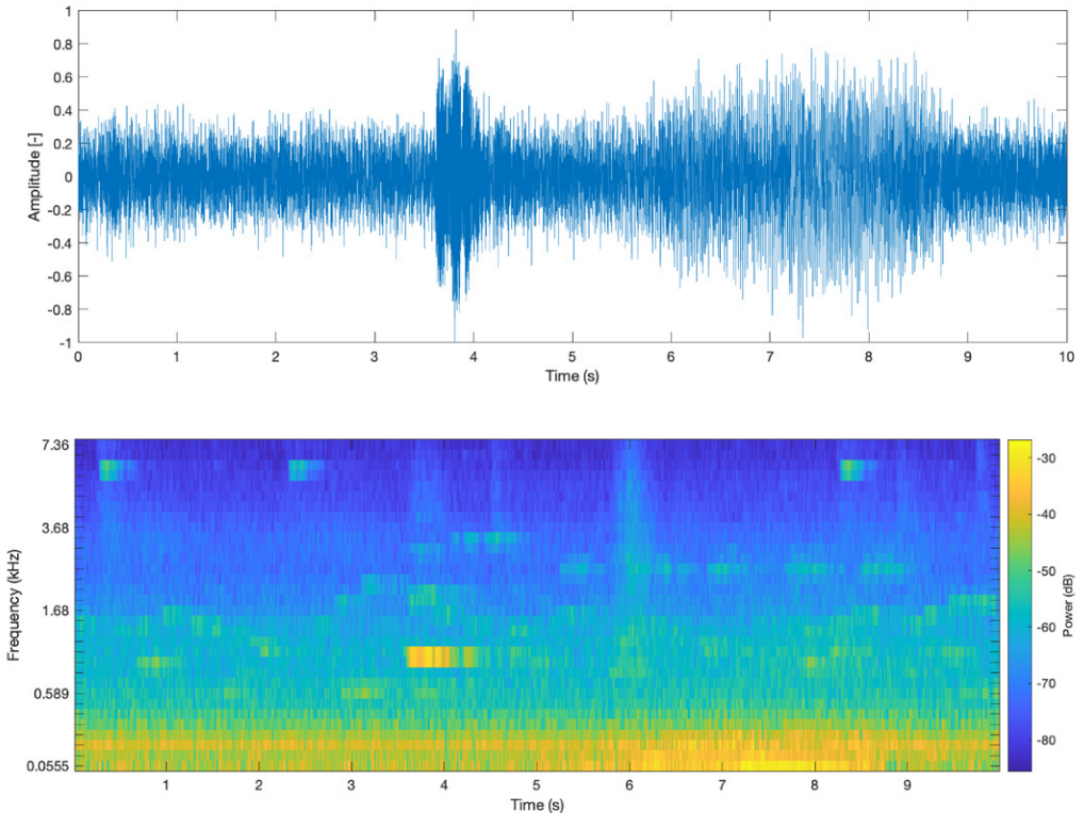



Figure 2.3: Example of a noisy signal and its STFT [5]

### 2.3.5 Wavelet Transform

The Wavelet Transform shares similarities with the Fourier Transform as both methodologies strive to express a given function using known functions. In the case of the Fourier Transform, the known functions are sines and cosines, which are non-local and stretch out to infinity. This fact causes the Fourier Transform to approximate sharp spikes very poorly. Wavelet analysis uses functions that are contained in finite domains, which are better suited for approximating data with sharp discontinuities [30]. In Figure 2.4, two functions are displayed, also referred to simply as wavelets, namely the Ricker and Morlet Wavelets. These examples were generated using the *SciPy* Python package with the *scipy.signal.morlet* and the *scipy.signal.ricker* method. Using different Wavelets to calculate the Wavelet Transform produces different results, so which type is selected influences the subsequent analysis.

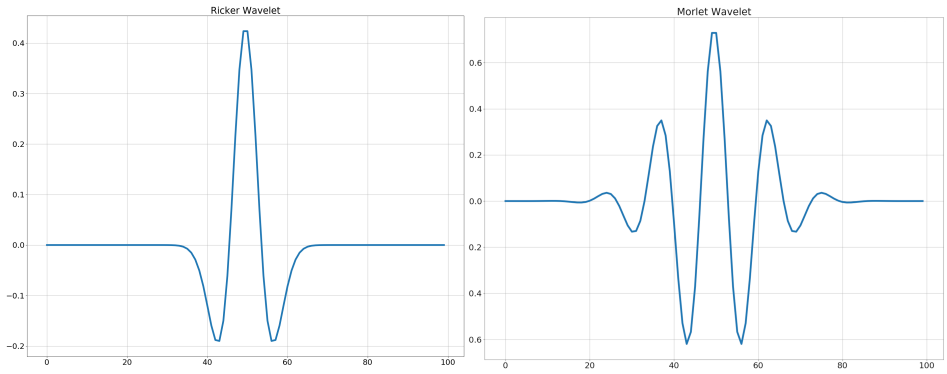



Figure 2.4: Ricker and Morlet Wavelets generated through *SciPy*

Using the Wavelet Transform, it is possible to determine how the frequency of a signal changes over time, while the Fourier Transform provides the prevalent frequencies of a signal, independently of its duration. In Figure 2.5, a signal that increases its frequency over time was generated through *SciPy* and the Wavelet Transform was computed.

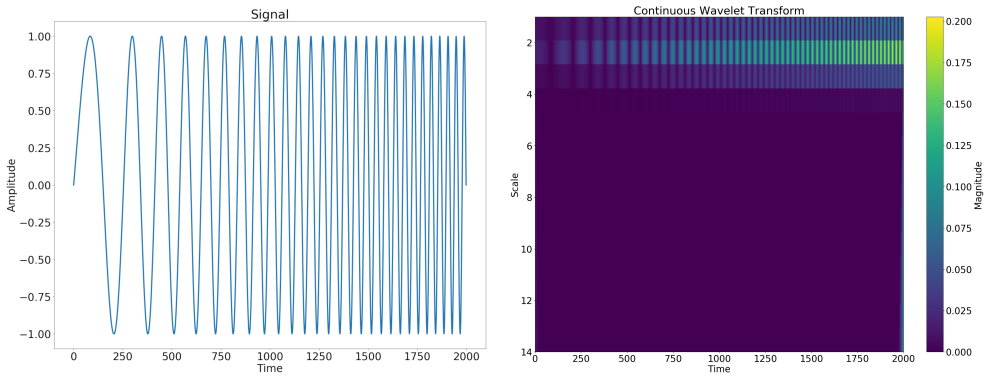



Figure 2.5: Signal with increasing frequency over time and its Wavelet Transform

### 2.3.6 Empirical Wavelet Transform

The Empirical Wavelet Transform is a recent advancement in signal processing and data analysis that builds upon the traditional Wavelet Transform. It is a data-driven signal processing technique that derives wavelet functions directly from the data itself, making it adaptive to the signal's specific characteristics. In contrast, the traditional Wavelet Transform uses predefined wavelets, which may not be as well-suited for non-stationary signals [31]. It outperforms other methods, such as the Empirical Mode Decomposition (which will be explored in Subsection 2.3.7) and the methods that derive from it, in theoretical reliability, meaning it has a solid mathematical foundation and no mode mixing phenomenon (also explored in Subsection 2.3.7). However, the Empirical Wavelet Transform has some disadvantages, including being considerably affected by high amounts of noise, being sensitive to modulation effects, requiring a great amount of parameter tuning and being computationally intensive, especially for larger datasets due to its data-driven nature [31].

### 2.3.7 Empirical Mode Decomposition

Empirical Mode Decomposition (EMD) is a signal processing technique that provides a data-driven way to decompose a complex signal into a set of simpler components known as *intrinsic mode functions* (IMFs). EMD perceives a signal as consisting of a fast oscillation superimposed on a slower one. Once the fast oscillation is extracted, the process treats the remaining slower component as the new signal and again regards it as a fast oscillation superimposed on a slower one. This iterative cycle continues until a predefined stopping criterion is met. While EMD does not employ fixed functions, as seen in wavelets, its fundamental approach shares conceptual similarities with the wavelet method. Both techniques involve breaking down the signal into its constituent details and approximations, followed by further decomposition of the approximation into details and an approximation [32].

Figure 2.6, taken from [33], displays a non-stationary signal, labelled *Data*, and its decomposition into several IMFs, which enables the extraction of information about the signal's time and frequency characteristics. This method is especially useful for non-linear and non-stationary data since it is self-adapting according to the analysed signal. Nonetheless, EMD possesses a few downsides, which include a lack of a sound theoretical basis and mode mixing (when an IMF obtained through the EMD decomposition exhibits components with varying frequencies) [31]. Other methods have been created which try to supplement EMD, Ensemble Empirical Mode Decomposition, Local Mean Decomposition and Empirical Wavelet Transform (in Subsection 2.3.6), to name a few.

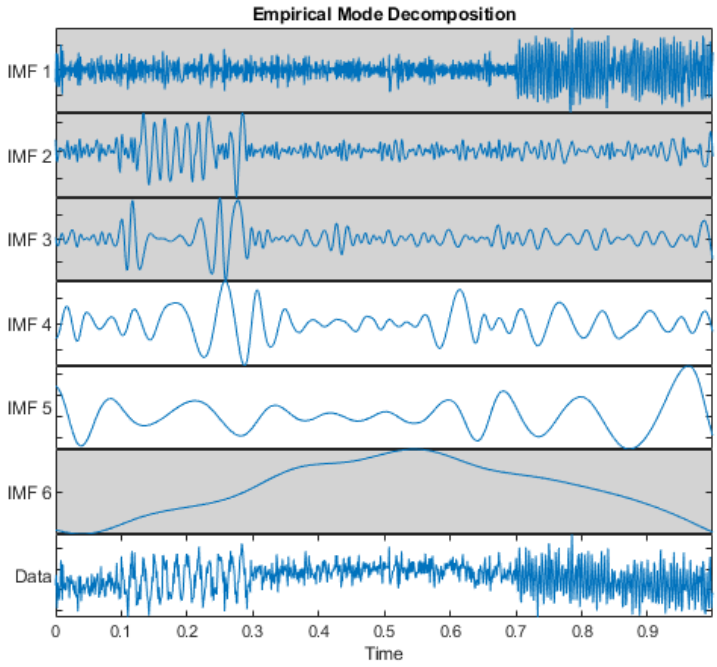



Figure 2.6: Example Signal and its decomposition into other functions through Empirical Mode Decomposition, from [33]

### 2.3.8 Hilbert Transform

The Hilbert Transform is a mathematical operation used in signal processing and mathematics to analyse and manipulate signals and functions, particularly in the context of time-domain and frequency-domain analysis. The key feature of the Hilbert Transform is its ability to create a complex-valued signal from a real-valued one, where the real part corresponds to the original signal, and the imaginary part represents a 90-degree phase-shifted version of the signal. This complex-valued signal is known as the analytic signal [26]. Equation 2.2 describes the Hilbert Transform, $x_h(t)$, which can be thought as the convolution between $x(t)$ and $h(t) = \frac{1}{t\pi}$.

$$H[x(t)] = \frac{1}{\pi} \int_{-\infty}^{\infty} \frac{x(\tau)}{t-\tau} d\tau \tag{2.2}$$

It can be used to extract the envelope and instantaneous frequency of a signal, which are useful for analysing non-stationary and nonlinear signals. The Hilbert transform can also be used for time-frequency analysis, phase analysis, and amplitude modulation detection [34].

### 2.3.9 Hilbert-Huang Transform

The Hilbert-Huang Transform (HHT) is a signal processing method that includes Empirical Mode Decomposition (EMD) and the Hilbert Transform. It is based on the local characteristic time scales of a signal and can decompose a complicated signal into a number of IMFs. The IMFs are extracted in a data-driven manner, which means they adapt to the signal's inherent dynamics [35].

The uniqueness of HHT lies in its ability to capture both time and frequency domain information simultaneously. Each IMF encapsulates a particular frequency component of the signal, and these frequency components are not fixed to the sampling frequency but rather adapt to changes in the signal itself. This adaptability makes HHT well-suited for analysing nonlinear and non-stationary signals [36].

The HHT exhibits similarities to the previous time-frequency analysis methods, as they share a common objective. Nevertheless, the HHT tends to be employed in scenarios where no previous knowledge of the signal exists [36]. When utilising the HHT, the other methods are also typically used and compared to check which approach is best for the situation at hand.

# Chapter 3

# Related Work

This chapter provides a comprehensive review and analysis of existing literature, research, and studies relevant to the dissertation's subject matter. The main research topics revolve around what components are most often analysed, the types of data used in PdM, Machine Learning methods typically employed in the field, preprocessing techniques, signal processing methods and hardware components and solutions matching the dissertation's goal.

## 3.1 Critical Components

Critical components encompass the essential elements within a system or process that are crucial to its proper functioning, efficiency, reliability, and success. In many cases, the breakdown of a single critical component can have severe consequences, leading to system failures, downtime, or even catastrophic events [1].

These components can be identified by analysing the frequency of faults in each component relative to the downtime caused by those faults [37]. When dividing components using this method, four quadrants can be described that display the most relevant information for the identification of the critical components of a production line. Figure 3.1 was adapted from [37] and illustrates the existence of these four regions using a case study from a production line that used wheeled mobile robots which recorded the number of faults of each type of component over one year and the average downtime each fault caused in the production line. For the case study, the boundaries for the quadrants were defined at 10 minutes of average downtime and 2000 faults in one year.

The four quadrants characterise the behaviour of components in the context of failure frequency and downtime, offering an insight into their economic efficiency and sustainability considerations.

- **1st Quadrant**: components that fail often and cause a high amount of downtime. These represent the most wasteful components, from an economic and sustainability standpoint, and, therefore, should be avoided at the machine's design stage.

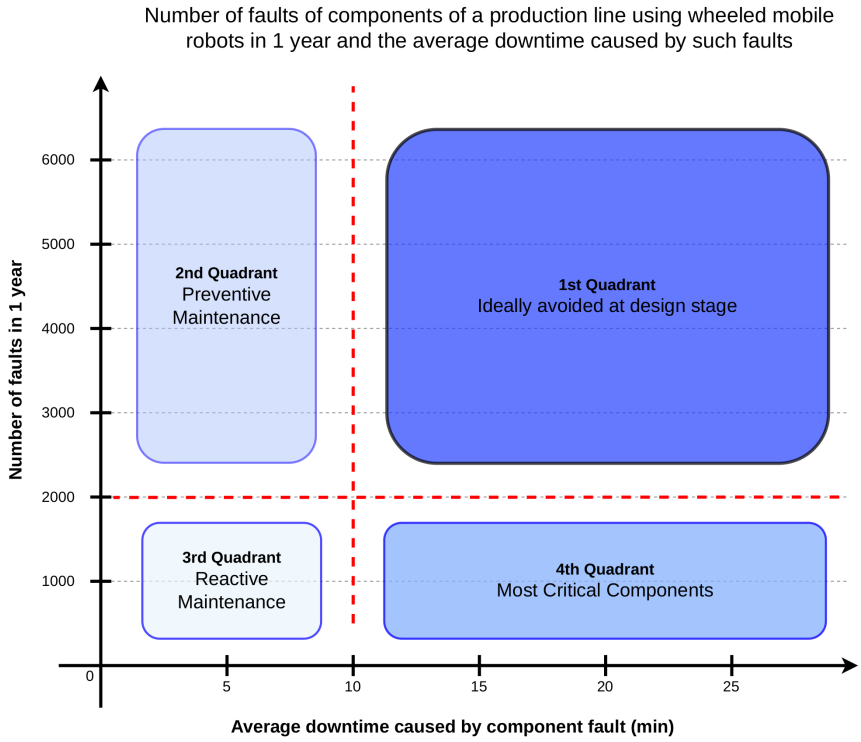



Figure 3.1: Number of faults in a production line in 1 year and average downtime caused by such faults, adapted from [37]

- **2nd Quadrant**: components that fail often but require a relatively low amount of time to be replaced. According to [1], can be maintained using PM without significant production efficiency losses.
- **3rd Quadrant**: components with infrequent failures, resulting in minimal downtime. Their impact on production efficiency is relatively low, making them suitable for RM strategies.
- **4th Quadrant**: components that seldom fail but demand extended downtime when faults occur. Components that meet these criteria are considered *critical components*. The rare faults do not justify replacing these components in the design stage, meaning their impact has the biggest impact in terms of potential downtime of all parts if third quadrant components were avoided at the design stage.

Components that belong to the fourth quadrant tend to be of high mechanical complexity, such as gearboxes [38] and induction motors [39], but low mechanical complexity critical components also exist in machinery, e.g. bearings [40] and gears [41]. Critical components are the best candidates for PdM at this moment because this type of maintenance strategy is still under development, and predicting faults in these components brings the biggest benefit to production lines. This is reflected in articles whose subject matter is condition monitoring, fault detection or anomaly detection, which tend to focus on monitoring the same kinds of components. [1] reviews articles regarding machine health and lists the monitored components, and its findings show that

the most frequently monitored components are the ones referring to rotary components and machining tools and processes, comprising 78% of a total of 93 analysed articles. Gears, bearings, and shafts represent the most frequently analysed single parts for machine health assessment in the rotary components group. Bearing faults account for 40% of rotating machinery fault events [31], which coincides with the literature, as most articles in the dissertation's area of interest account for fault detection of bearings. However, it is worth noting that induction motors and gearboxes are integral components of virtually all industrial machines and hold critical importance. As a result, they are consistently subjected to condition monitoring and fault detection, as evidenced by [42] and [43].

## 3.2 Data Collection

Data collection plays a critical role in condition monitoring and PdM, processes that involve continuously monitoring the health and performance of machinery and equipment. Effective data collection is essential for accuracy, enabling the analysis of various parameters and signals to identify patterns, trends, and anomalies. It represents the foundation for developing predictive models and algorithms that provide early warnings, enhance equipment reliability, and ultimately drive operational efficiency.

### 3.2.1 Vibration Analysis

Vibration analysis is the most widely used approach for PdM. It is primarily applied to rotating machinery, either continuously or at predetermined intervals, to gather insights into the system's current condition [25]. Numerous vibrations are closely associated with periodic events in the machine's operation, such as rotating shafts, meshing gear teeth, rotating electric fields, and more. The frequency at which these events repeat serves as a direct indication of their source, making frequency analysis the foundation of many robust diagnostic techniques [44].

This form of analysis for machine condition monitoring has a number of advantages compared with other methods. In contrast to temperature or oil analysis, which will be reviewed further along, it reacts immediately to change, which means it can be used for permanent as well as intermittent monitoring. This approach allows maintenance to be performed more rapidly on a critical component, leading to a decrease in cost as there will be less downtime. Certain components can be repaired before reaching a state of irreparable damage, extending their useful life. Another advantage lies in the ability to extract subtle fault indications from background noise and other masking signals in vibration signals through the application of signal processing techniques [44], a topic that will be explored further ahead.

For measuring vibration, two types of measurement exist: absolute and relative vibration. Absolute vibration refers to measuring vibration levels with respect to an absolute reference point outside of the system itself. Transducers that capture this type of vibration are either velocity, e.g. seismometers, or acceleration transducers, e.g. accelerometers. Relative vibration is, as the name implies, the measurement of vibration relative to a point within the system under measurement,

which can captured through the use of proximity transducers [44]. Accelerometers have emerged as the predominant transducer in condition monitoring research. This preference arises from their ease of installation in comparison to proximity probes, which necessitate closer proximity to the specific machine part under inspection, such as a rotating shaft. Proximity probes may require drilling into certain components or other costly operations. In contrast, accelerometers can be conveniently placed in more accessible locations while still delivering precise and reliable readings. Proximity probes are also susceptible to other outside factors such as dust, dirt and thermal expansion [40]. This does not mean that accelerometers are immune to noise or impervious to environmental factors. The accuracy of their readings can be influenced by the mounting method used, meaning that careful consideration of the installation is crucial to ensure reliable measurements [45].

Researchers exploring novel signal processing or Machine Learning techniques for fault detection with vibration data benefit from the availability of reliable datasets that have been extensively employed in the field. Among these datasets, the Case Western Reserve University Dataset [46] holds a prominent position, to the extent that there are even reviews dedicated to the algorithms applied to this dataset, such as [47]. This review focuses on how the dataset was produced, while comparing it to other well-known ones, e.g. the Paderborn University Bearing Dataset [48] and the PRONOSTIA dataset [49], and reviewing Deep Learning algorithms that employ these.

In [50], a testing platform consisting of a motor, shaft and load is created and three accelerometers are used to record vibration data for detecting unbalance of the shaft. The data is used as input for a Convolutional Neural Network while testing different preprocessing techniques. The best classification model obtained 98.6% accuracy in identifying four different degrees of misalignment. [51] relies on thresholds of vibration for classifying faults according to their severity of a conveyor belt motor, based on a theoretical model built specifically for the motor's conditions, including power and rotation speed.

Additional articles that utilise vibration as its source data to either detect or diagnose faults will be reviewed throughout the rest of this chapter, but the focus of the review will shift to the techniques used to process the data.

### 3.2.2 Acoustic Analysis

Under typical operating settings, most machines produce consistent sound patterns, which can be recorded as reference. Changes in the reference pattern indicate wear or other types of component deterioration [25]. At a low-frequency range (which is often considered up to 400 Hz), unbalance, misalignment and other mechanical construction shifts can be detected [52]. Other types of failures can be identified at higher frequencies, such as gearbox faults and bearing faults [53] [54].

Using sound data as a means of condition monitoring has two distinct advantages over vibration analysis: fewer costs associated with sensor installation in preexisting machinery [5] and the lack of physical interaction with the test subject, which does not affect the system's vibration dynamics [55].

According to [56], the most popular datasets used in acoustic condition monitoring up until its publishing were: ToyADMOS [57] and MIMII [58]. The ToyADMOS dataset comprises three subsets of equipment audio samples in normal and anomalous operating conditions. The subsets were meant to replicate different scenarios: product inspection, fault diagnosis for a fixed machine and fault diagnosis for a moving machine. The recordings were of a toy car, toy conveyor and toy train, supplemented with controlled environmental noise, to reproduce real scenarios more accurately. The microphones used were the SHURE SM11-CN, whose frequency response is charted between 50 Hz and 15 kHz [1]. Another iteration of this dataset, ToyADMOS2 [59], has been since published. It complements the first version by possessing controlled domain shifts and higher variation (machine models and operating speeds) in the same operating conditions (normal and anomalous). *Domain shifts* refers to a change between the training dataset used for an algorithm and the dataset it encounters during deployment. The article also adds a different microphone model (TOMOCA EM-700) whose frequency response is charted from 20 Hz to 20 kHz [2]. The MIMII dataset consists of audio recordings sampled in normal and anomalous working conditions of four types of machines: valves, pumps, fans and slide rails. Each machine's anomalous recordings included different root causes, e.g. for the pump, a leakage state was recorded separately from a clogged state. The recordings were made using a TAMAGO-03 (with a frequency response charted up to 16 kHz [3]) microphone array, which enables multi-channel-based analysis as well as the more common single-channel-based analysis.

Most articles which use sound data typically use microphones with a charted frequency response charted up to 20 kHz, as is the case with all articles which use the aforementioned datasets, e.g. [60] [61]. Other implementations that exemplify the previous statement are [62], using the ReSpeaker 4-Mic Array with SPU0414HR5H-SB microphones (up to 10 kHz), and [63], using the GRAS 40PH-10 microphone (up to 20 kHz). However, acoustic analysis can go beyond human hearing and into ultrasonic analysis. Ultrasonic sensors can be used to detect other types of faults, e.g. leakage from water pipes [64] and railway inspection [65]. Ultrasonic analysis is commonly referred to as Acoustic Emission analysis, which is further explored in the next Subsection.

### 3.2.3 Acoustic Emission Analysis

Acoustic emission is a phenomenon defined as the propagation of transient stress waves as a result of localised internal energy, such as a micro-fracture in elastic material [66]. Acoustic emission waves are not sonic waves but elastic waves in a solid whose frequency range goes beyond the human audible range (usually up to 20 kHz [67]). Sonic waves higher than the audible range are defined as ultrasonic waves. Acoustic waves, strictly speaking, are neither ultrasonic nor acoustic and are commonly detected up to 1 MHz [68].

These emissions can be monitored to detect micro-fractures in many different scenarios, more commonly applied to structural health monitoring by analysing defects in concrete [69] [70]. It

[1] https://pubs.shure.com/guide/SM11/en-US
[2] https://www.tomoca.co.jp/brand/tomoca/etc_mic/em-700/
[3] https://www.sifi.co.jp/en/product/microphone-array/

is also used to monitor seismic activity in scenarios such as earthquake detection [71]. The most relevant use cases are those applied to machine health monitoring, such as bearing condition monitoring [72] [73] and rotating machinery monitoring [74], including gearboxes [75].

### 3.2.4 Temperature Analysis

Thermography refers to the use of a non-invasive imaging technique that utilises infrared technology to capture and visualise the temperature patterns of objects and surfaces. By detecting and measuring the emitted heat radiation, thermography produces thermal images that depict variations in temperature across a specific area. It has been used in several different research fields, including medical imaging, vulcanology and chemical process control, but the most relevant use case scenario for this project is in condition monitoring.

Any surface can release energy as electromagnetic waves through the radiation process, which is the foundation of this technology. Objects emit electromagnetic radiation in the infrared region of the electromagnetic spectrum when their temperature exceeds 0 K [76].

In condition monitoring, this technique has been extensively adopted in electrical equipment inspection, material fatigue evaluation, weld inspection and printed circuit board (PCB) inspection, among other varied uses [76]. The biggest advantages of using this technology are the fact that it eliminates the need for direct physical contact between the Thermography sensor and the examined component and that several different points of a surface and objects can be measured simultaneously. These features enable checking a large surface area to check for hotspots or other temperature-related anomalies [77].

Faults in mechanical and electrical components can lead to a temperature increase in the affected component [76], which makes it a feature suitable for condition monitoring and, therefore, PdM. The most common use of temperature analysis is through Infrared Thermography (IRT) [1], which utilises the principle that objects emit infrared (IR) when their temperature exceeds absolute zero and that the wavelength of the light grows as the object's temperature increases. This can be used to visualise temperature changes by displaying different temperatures as different colours, as displayed in Figure 3.2, but it must be noted that in addition to temperature, surface characteristics, object shape and view angle also affect the amount of radiation that reaches the thermography device [25]. Since thermography merely relies on detecting radiation produced from the component under examination, direct contact with the component is not necessary [77], which reduces the impact the sensor has on the health signals of the examined component. As opposed to manual inspection, automatic IR-based condition monitoring does not require a human expert to interpret the thermal images [78], meaning models can be made to automatically detect faults or even predict them.

### 3.2.5 Oil Analysis

Oil Analysis is a non-intrusive method that involves periodically sampling and analysing lubricating oil or hydraulic fluid to detect potential signs of wear, contamination, and degradation within

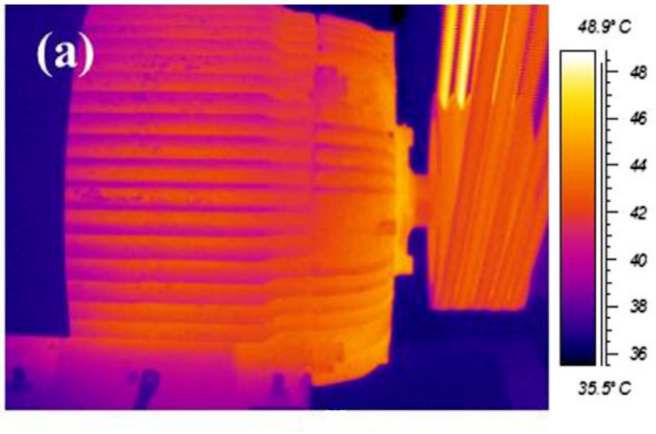



Figure 3.2: IRT Image of an AC Motor from [76]

the machinery. By examining the oil's physical and chemical properties, as well as the presence of metal particles, contaminants, and additives, oil analysis provides valuable insights into the condition of critical components such as bearings, gears, and hydraulic systems [25]. Anomalies detected through oil analysis can help identify early signs of equipment deterioration, abnormal wear patterns, and potential failures, enabling proactive maintenance interventions to prevent costly breakdowns and unplanned downtime.

### 3.2.6 Electric Current Analysis

Electric Current Analysis relies on monitoring the electrical current consumption of different elements of a machine to check if a fault has occurred. It is especially helpful in identifying electrical faults in components such as motor drivers, windings and contactors. Faulty components produce high currents and winding overheating, which may lead to irreversible damage, making Electric Current Analysis a valuable tool for preventing such deterioration [79]. Electric Current Analysis is a frequently employed technique, especially in the context of electric motors, due to its versatility in monitoring multiple components, particularly through the precise examination of motor current signals. This specialised approach is exemplified in [80], where the current signal of a permanent magnet synchronous machine is harnessed to successfully identify three distinct types of faults within the system.

## 3.3 Prediction Algorithms

Prediction algorithms play a crucial role in the field of PdM, where the goal is to anticipate equipment failures and perform maintenance activities proactively, reducing downtime and optimising resource allocation.

### 3.3.1 Prediction Problem Definition

This project's ultimate objective is detecting looming faults in the AGV, possibly discerning which components are about to experience faults. However, due to time constraints, this dissertation only plans on determining if a fault is about to take place without discriminating if it will affect the motors or roller bearings or the electrical wiring. In the field of data analysis, this is a problem known as *Anomaly Detection* defined as "finding patterns in data that do not conform to expected behaviour", according to [16]. This definition fits the problem at hand since impending faults can be predicted from sensor measurements that deviate from the "normal" readings, e.g. the increase of a roller bearing's vibration frequency and the temperature increase from a motor driver's heat sinks. A noteworthy characteristic feature of Anomaly Detection is that using it enables detecting faults for components that are not modelled or that do not possess previously researched fault diagnosis methods. As the case with planetary gearboxes, whose behaviours differ significantly from fixed axis gearboxes, which invalidates using functioning fault diagnosis methods from the latter to be used with the former, as noted by [43].

Anomaly Detection can be approached through two primary methodologies: statistical methods and Machine Learning algorithms. Statistical methods rely on theoretical insights into the data or utilise simulated models with known behaviour to identify anomalies. In contrast, Machine Learning (ML) methods do not rely on pre-existing knowledge and can autonomously determine anomalies within the dataset. Given the absence of a comprehensive model for the AGV based on theoretical principles and analysis, the utilisation of ML techniques emerges as the most fitting approach within the context of this dissertation.

It is important to note that Anomaly Detection represents a *classification* problem since its primary objective is to determine whether a set of measurements at a particular sample time falls into one of two categories: anomaly or non-anomaly. Before selecting what ML algorithms to implement, there is a need to better define the problem at hand, taking into account the lack of a similar public dataset that encompasses all features of the planned sensor module: vibration, temperature and sound data. A custom dataset must be created for evaluating the performance of different algorithms.

ML algorithms for Anomaly Detection can be broadly categorised into three groups, each of which aligns with the availability of labelled data. These groups, as outlined by [81], are Supervised Learning, Semi-Supervised Learning, and Unsupervised Learning:

- **Supervised Learning** involves a dataset in which all classes are explicitly represented. For instance, in the case of temperature data, a label might indicate that 75.6°C corresponds to the anomaly class, while 60°C belongs to the non-anomaly class.
- **Semi-Supervised Learning**, on the other hand, entails a dataset where not all classes are clearly delineated or represented.
- **Unsupervised Learning** operates with data that lacks predefined classes, and the algorithm's task is to autonomously identify patterns and anomalies without explicit labels.

Considering the project's ultimate goal of deployment in an industrial setting, noise is a significant problem. It cannot be considered white noise since distinctive patterns can emerge from characteristic machinery parts, such as fans. As previously seen, in the MIMII dataset [58], the equipment sound recordings were mixed with background noise from multiple factories to simulate a real environment. In the spectrograms available in the article, it is possible to observe patterns that do not correspond to either the equipment subject to recording or white noise. The fact that industrial noise is not Gaussian is rather intuitive as multiple concurrent activities occur during working hours, e.g. material transportation from and to different stations and changes of the cycles of production. One way to combat this is to use other types of measurements that are not affected by background noise to reinforce the built model to better resist noise. An example of this type of measurement is *temperature*, considering that a component's temperature is not influenced by background acoustic noise in a meaningful way. More sophisticated techniques to reduce noise will be explored further along.

### 3.3.2 Statistical Methods

Statistical methods in this context combine signal processing techniques with theoretical knowledge for fault detection. Such is the case with [82], whose objective is to identify faults in roller bearings using a Microelectromechanical Systems (MEMS) microphone contained in a small battery-powered module. It records sound in 1-second snippets and converts the audio from the time domain into the frequency domain using the FFT algorithm combined with a windowing function to limit the range of the FFT.

$$BPFO = \frac{Zf_r}{2}\left(1 - \frac{d}{D}cos\alpha\right) \tag{3.1}$$

In Equation 3.1, BPFO refers to the Ball Pass Frequency of the Outer Raceway of a rolling element bearing [4], which describes the resonant frequency of the outer shell of a rolling element bearing based on its physical characteristics. $Z$ is the number of rolling elements, $f_r$ is the inner rotation frequency, $d$ is the rolling element diameter, $D$ is the pitch diameter and $\alpha$ is the contact angle. Figure 3.3 displays a schematic drawing of the basic structure of a ball bearing. It is worth noting that some variables have a different name from Equation 3.1 since the drawing was taken from a different source. Other resonant frequencies are used, such as the one for the Inner Raceway and for the Ball Spin Frequency, which have their own equations based on the same characteristics. [82] calculates these frequencies for the tested bearings and sets a small frequency window due to possible deviations from the ideal model. The proposed model, as previously detailed, turns the audio sample into the frequency domain and checks the amplitude of the signal in the previously calculated frequency windows, and if the amplitude surpasses a certain threshold, there is a mechanical fault in that specific part of the bearing, such as the outer race. It is important

[4] A rolling element can be either a sphere or a cylinder

to note that the algorithm needs calibration to set the minimum threshold since it is highly affected by background noise.

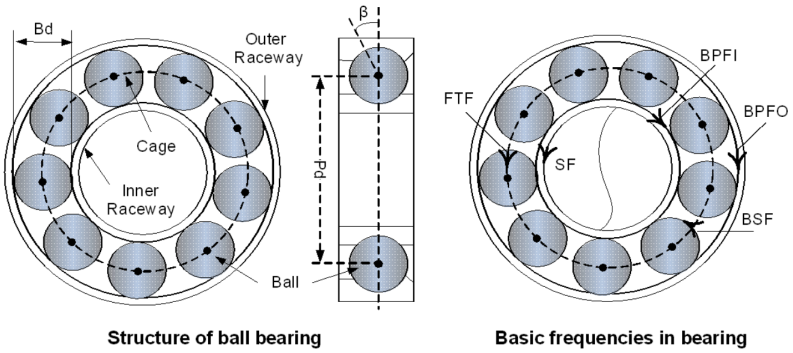



Figure 3.3: Ball Bearing Internal Structure, from [83]

### 3.3.3 Supervised Learning Methods

Supervised Learning for classification tasks is based on the premise that the training data contains instances of all categories, in this case, normal behaviour and all expected anomalies. Many articles which focus on fault diagnostics tend to use Supervised Learning Methods due to potentially producing more accurate predictions provided that enough data for training is available [16].

#### 3.3.3.1 Decision Trees

Decision trees are a widely used and intuitive machine learning algorithm for classification and regression tasks, presenting a flowchart-like structure where internal nodes represent features and leaf nodes represent class labels or predicted values. The goal is to create a model that predicts the target variable by learning simple decision rules from the data. Each tree can be seen as a "piecewise constant approximation," partitioning the feature space into regions with unique predictions. The interpretability of decision trees allows for better understanding and insights into the model's decision-making process [23]. Due to the latter feature, it is a popular choice of algorithm for Anomaly Detection. However, it is prone to overfitting and more efficient algorithms that achieve better results with a similar structure, e.g. Random Forest has overtaken Decision Trees as a choice in more recent works [81].

#### 3.3.3.2 Random Forest

The Random Forest (RF) algorithm combines multiple Decision Trees in order to increase predictive accuracy and reduce overfitting [24]. By aggregating the predictions of individual Decision Trees, the RF algorithm mitigates the tendency of any single tree to make biased or erroneous predictions. Additionally, the term "Random" refers to the fact that this algorithm introduces an

element of randomness during both tree construction and prediction, which enhances the model's robustness.

In [84], the condition of a friction stir welding tool is monitored using vibration signals, where several Decision Tree-like algorithms are compared, including the RF algorithm. Multiple states of the tool were recorded, in normal condition, in partially dysfunctional condition and in broken condition, at two different speeds. When comparing four different ML algorithms, RF obtained the highest accuracy for classifying the condition of the tool based on the testing data, obtaining a 73.16 % score for the 1400 rpm test and a 93.51 % score for the 1800 rpm.

#### 3.3.3.3 Gradient Boosted Decision Trees

Boosting algorithms, in general, combine weak learners, as in models slightly better than random, into stronger learners in an iterative way. In the case of Gradient Boosted Decision Trees, the weak learners are Decision Trees [85]. Gradient Boosting refers to the way the loss function is computed, relying on gradient descent optimisation. The most popular Gradient Boosted Decision Trees frameworks for this variety of ML algorithm are XGBoost [86], LightGBM [87] and CatBoost [88]. These are widely spread across multiple fields of study since they can be used in both classification and regression tasks. According to [85], LightGBM provides the fastest training times, while CatBoost had the most accuracy for classification tasks. XGBoost produced the most balanced results. All tests considered hyperparameter tuning, as the results might have been fairly compared without adjusting each model for the best results possible.

In [89], a wind turbine is simulated, as well as several fault states, including faults in the rotor, controller and pitch actuator. A model using XGBoost is created in order to diagnose the simulated faults after a feature analysis was performed to denote which features were the most relevant to use as input. The article concludes that the created model was adequate for diagnosing faults in wind turbines, including offshore ones, in several different working conditions. It was also shown that the model performed better than Random Forest and an SVM which were also tested.

#### 3.3.3.4 Artificial Neural Networks

Artificial Neural Networks (ANN) are a very popular ML model archetype, based on the structure of biological neural networks. It consists of a network with a large number of simple processors with many interconnections in parallel. Rather than adhering to rules defined by human experts, ANNs acquire fundamental principles by learning from a set of symbolic situations provided in examples given in the training data. Since ANNs are able to learn solely from examples, ideal for random data, fuzzy data, and nonlinear data, variations of it are widely used in the field of Condition Monitoring [90], due to some systems' high complexity and nonlinear behaviour. In [50], which as been previously reviewed, a Convolutional Neural Network (CNN) is used to detect if a shaft is unbalanced through vibration data. The article compares using raw sensor data through the CNN and preprocessing the data before using it as input for the CNN. Although the CNN using preprocessed data performed significantly better than the raw data CNN, the latter still achieved

an average of 79% accuracy across all tests, which demonstrates that ANNs and their variations can obtain satisfactory results from crude data. However, it is demonstrated that preprocessing the data before inputting it into a CNN can lead to substantial improvements in the performance of the model, given that the accuracy raised to 97.5 %.

#### 3.3.3.5 Support Vector Machines

The Support Vector Machine (SVM) algorithm aims to find an optimal hyperplane that best separates different classes, maximising the margin between them. SVMs have been successfully applied to multiple applications ranging from speech recognition, and face detection to fault diagnosis. By using a *Kernel Trick*, SVM can effectively handle non-linearly separable data, which is the reason for being able to yield suitable results in such a varied span of research fields [24]. The objective in [91] is to diagnose faults in roller bearings, for which an SVM, in conjunction with Principal Component Analysis (PCA), is used. This method was chosen due to its non-linear regression abilities since non-linear data was used, vibration signals. When classifying different faults in the tested bearings, the lowest accuracy obtained was 96.6%.

### 3.3.4 Semi-Supervised and Unsupervised Learning Methods

Semi-supervised Learning offers a promising avenue for Anomaly Detection, particularly in scenarios where obtaining a large number of labelled anomalies is challenging. This approach leverages a combination of labelled normal samples and a more extensive set of unlabelled data, enabling the chosen model to learn meaningful representations of normal patterns [92]. Unsupervised Learning techniques in the realm of Anomaly Detection possess the broadest applicability due to the aforementioned fact of the rarity of availability of anomalous data while assuming that the testing data contains a significantly higher frequency of normal instances compared to anomalies [16].

#### 3.3.4.1 Isolation Forest

The Isolation Forest algorithm was specifically developed for Anomaly Detection. It explores a different approach from those typically used by other techniques to try to solve the Anomaly Detection problem: isolating the anomalies explicitly instead of profiling the normal instances [93]. The article that first proposes the algorithm, [93], considers isolation as "separating an instance from the rest of the instances" and anomalies as "few and different", meaning that anomalies are prone to isolation. It has successfully been used in the fault detection field, as in [94], where Isolation Forest was employed to detect faults in an aero-engine. For aeronautical scenarios, it is difficult to obtain actual fault data, so the model was trained in a Semi-Supervised manner, using only normal data. For testing purposes, anomalous data was simulated through a State Variable Model of the engine to mimic most fault states, e.g. faults related to the gas path (an aero-engine specific part), and through adding noise to the measurements to replicate the impact of vibration. Four different Anomaly Detection algorithms were used: Angle-based Outlier Detection, Local

Outlier Factor, One-class Support Vector Machine and Isolation Forest. Out of these, the most accurate and with the lowest running times was Isolation Forest.

#### 3.3.4.2 One-Class Support Vector Machine

One-Class SVMs differ from SVMs as their objective is to build a boundary between normal and anomalous data. The fundamental approach involves assessing each test instance to ascertain whether it falls within the learned region. If a test instance is found to lie within the learned region, it is classified as normal; otherwise, it is classified as anomalous [16]. This makes its primary use Anomaly or Outlier Detection. Figure 3.4 shows a One-Class SVM in a two-dimensional space, where the red line represents the boundary between normal and anomalous data points.

In [95], a One-Class SVM is built using 300 samples of a simulated system, which takes into account noise, instrument failure, formatting errors and non-representative samples, i.e. misreadings. The results of the implementation show that the One-class SVM was robust to outliers but still sensitive enough to determine if a test instance was anomalous or normal, demonstrating that this algorithm is adequate for Fault Detection.

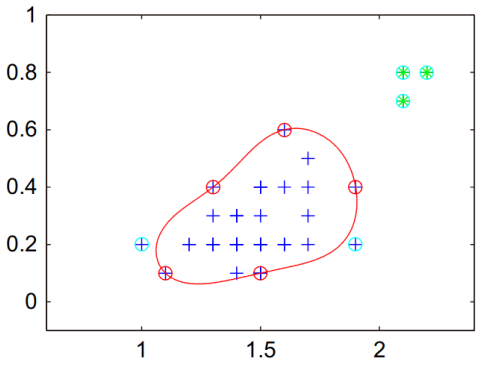



Figure 3.4: One-Class SVM Example in 2 Dimensions, from [95]

#### 3.3.4.3 Local Outlier Factor

The Local Outlier Factor is an unsupervised anomaly detection algorithm that assesses the abnormality of data points within their local neighbourhood. It calculates the degree of isolation of each data point by comparing its density to that of its neighbouring points. If a data point has a significantly lower density compared to its neighbours, it is considered an outlier. It is widely used in Anomaly Detection and as a comparison metric in articles comparing their proposed method to more traditional methods, such as the Local Outlier Factor. It is based on a concept of a local density, where locality is given by $x$ nearest neighbours, whose distance is used to estimate the density [16].

The Local Outlier Factor (LOF) is frequently employed as a benchmark when evaluating new algorithms. For instance, in [94], it falls into a category of traditional methods for anomaly detection, particularly within the realm of fault detection in engines.

#### 3.3.4.4 Autoencoders

Autoencoders are a type of neural network intended for feature extraction and data compression and are trained to attempt to copy their input to its output. The algorithm reconstructs its input data by learning a compressed representation of the input in a lower-dimensional space [92]. Figure 3.5 shows the basic structure of an Autoencoder: the input layer followed by encoding layers to reduce the dimensionality of the input, the layer containing the least amount of information that the model needs to classify the input, and decoding layers leading to the output which will try to replicate the input. During training, the Autoencoder is exposed only to normal data, and it learns to capture the essential patterns and features of the normal samples. Anomalies, being dissimilar to the learned normal patterns, result in higher reconstruction errors. This makes Autoencoders sensitive to anomalies and can effectively distinguish them from normal data. By setting a threshold on the reconstruction error, data points with higher errors are flagged as anomalies. In [96], a hybrid model is created that deploys an Autoencoder in conjunction with the STFT to diagnose faults in rotating machinery, namely a hydro generator rotor, using vibration signals, for which only normal functioning data was provided. The proposed method outperformed two other models used for comparison, a modified Autoencoder that uses Softmax Regression and a CNN, in terms of accuracy. In addition, the models were compared when trained using 20 up to 2000 labelled samples, with the hybrid method beating the other models at every stage.

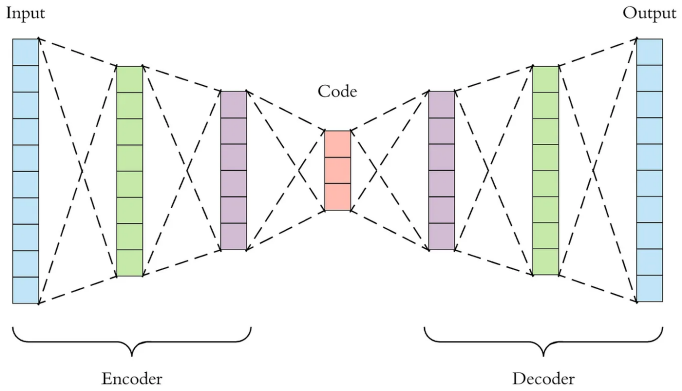



Figure 3.5: Autoencoder structure from [97]

## 3.4 Preprocessing

As with all statistical models, noise, missing values, and other types of disturbances affect the intended prediction. Certain techniques are employed to reduce the influence these characteristics

have on the expected results. The set of procedures to make the available data better suited for the intended use is called Data Preprocessing. Considering the given definition for the problem in Subsection 3.3.1, Anomaly Detection is the chosen approach to reach this project's objective. However, this technique bears an inherent uncertainty: the difference between anomalies and outliers [16]. Some articles do not bother to set these apart [16], but these tend to refer to the field of cybersecurity, where network packets that deviate from the norm might represent suspicious activity. In this context, separating anomalous behaviour and outliers is essential since the data under scrutiny is obtained from sensors that occasionally fail or provide a misreading, so a method to discern these cases is needed to identify actual anomalous behaviour. In the following Subsections, several data preprocessing techniques will be discussed, including noise reduction and outlier removal.

One aspect of preprocessing that will not be analysed is missing value imputation since one of the goals of this dissertation lies in building a testbed, meaning the datasets will be created specifically for this project and not sourced from other sources. This way, missing values can be avoided.

### 3.4.1 Outlier Removal

As previously stated, outliers in this context are regarded as misreadings from the sensors, so this view was taken into account when reviewing relevant articles for adequate techniques.

The Kalman Filter and its variations, e.g. Extended Kalman Filters, are a well-known method for removing noise from sensor readings. However, these rely on some presumptions that might not be guaranteed in the dissertation context: Gaussian sensor noise, a control signal and the system's dynamics [98]. Since one of the goals of this project is for the model to be agnostic to the system's theoretical dynamics, there is no available model and therefore, no control signal or system dynamics are known to be applied to a Kalman-like Filter. It is also not guaranteed that the sensor noise follows a Gaussian distribution since the sensors in the use-case scenario will be exposed to industrial environmental noise, meaning random occurrences might happen from the sensor's perspective.

Non-parametric filters, such as the Particle Filter, do not rely on Gaussian distributions for their sensor readings and can use other probability distributions, including using random sample points sampled at the time of deployment [98]. This type of filtering method is also not adequate for the problem at hand due to also relying on the system's dynamics.

Vibration and sound signals present similar characteristics, as already demonstrated, so analogous approaches can be applied to remove outliers. In [50], vibration data is used to detect imbalance in a rotating shaft through the use of ML algorithms. It compares the performance of a Convolutional Neural Network using the raw sensor data and preprocessed data and also with a Random Forest on Automatically Extracted Time-series Features. The most relevant aspect of this article, in the context of this Section, is the first appointed comparison since the preprocessed data achieved a better result. The method for preprocessing described in the article is the following: sampling 4096 values of vibration data equalling 1 second of data; converting the sensor

data to the frequency spectrum using the FFT; removing the median component from each Fourier coefficient and dividing the remaining values by the interquartile values. By analysing the available code [5], it is possible to verify that the former operations are executed using the *Scikit-Learn* method *sklearn.preprocessing.RobustScaler* [6]. In [99], the goal is to diagnose faults in a gearbox using the motor's (the one attached to the gearbox) current values and a CNN. It is important to note that these values are for Alternate Current, which share the oscillatory properties of vibration signals. Before using the current values, a second-order Butterworth low-pass filter with a cut-off frequency of 500 Hz is used to remove noise from the signal. However, this filter is only applied after removing outliers which are defined as more than three times the Scaled Median Absolute Deviations ($MAD_S$) characterised in Equation 3.2, where $A$ is the input data and $C$ is defined in Equation 3.3.

$$MAD_S = C \cdot median(|A - median(A)|) \tag{3.2}$$

$$C = \frac{1}{\sqrt{2} \cdot erfcinv(\frac{3}{2})} \tag{3.3}$$

In Equation 3.3, the term $erfcinv$ is the inverse of the Gauss error function, $erf$, which is described in Equation 3.4.

$$erf(x) = \frac{2}{\pi} \int_0^x e^{-t^2} dt \tag{3.4}$$

Other types of data used in PdM, such as temperature and direct current, are not waves, and their measurement readings do not oscillate significantly between samples taken milliseconds apart as in the case of sound measurements. This means that more standard outlier removal techniques can be applied to these data types. Assuming the sensor readings follow a Gaussian Distribution, as they are not subject to as many noise sources as their vibration counterparts, parametric techniques can be used [16]. One of the simplest is to declare that all measurements that are more than $3\sigma$ (standard deviations) away from the distribution mean $\mu$ are outliers since the $\mu \pm 3\sigma$ region contains 99.7% of the data instances. More sophisticated algorithms are used, such as the Local Outlier Factor [16], which has been reviewed in Subsection 3.3.4.

## 3.5 Digital Signal Processing

The Signal Processing techniques explored in Chapter 2 are exhaustively used throughout the literature. Additionally, articles that have already been reviewed make use of them to enhance the

[5] https://github.com/deepinsights-analytica/ieee-etfa2020-paper
[6] https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.RobustScaler.html

performance of fault-detecting or diagnosing algorithms. The following Section will focus on the application of Signal Processing methods and their diverse use cases in augmenting performance.

### 3.5.1 Discrete Fourier Transform

As seen in the previous chapter, the DFT is widely used in the current technology landscape, and the field of condition monitoring is no exception. Frequency analysis is essential to several different model-based and data-driven fault detection approaches, whether in terms of vibration, acoustic emission or sound analysis, since these physical quantities are generally harder to analyse using time-series data [29].

#### 3.5.1.1 DFT in Model-Based Analysis

Frequency analysis is at the basis of most articles about condition monitoring that use vibration and sound as measurements. [100] uses the FFT algorithm to create a model to monitor ceramic ball bearings in a starved lubrication state based on simple thresholds. Empirical Mode Decomposition (EMD) is also extensively used in Model-Based methods, such as the case with [101] where EMD is used in fault detection for roller bearings. In [102], EMD is adopted to de-noise vibration measurements before using them to train a Probabilistic Neural Network. [31] implements the use of the Empirical Wavelet Transform (EWT) to build the frequency spectrum of bearing vibration data and uses a threshold to diagnose the nature of the fault of the affected bearing.

#### 3.5.1.2 DFT in Data-Driven Analysis

[54] uses the common FFT algorithm in order to calculate the Power Density Spectrum, the Root Mean Square Frequency and the Frequency Center to analyse vibration and sound data to detect faults in a worm gearbox using an ANN. In [103], vibration measurements are put into the frequency realm using FFT and fed into a predictive maintenance model with a Convolutional Neural Network to diagnose faults in rotating machinery.

### 3.5.2 Short-Time Fourier Transform

The STFT is often used in combination with machine learning algorithms to create models capable of distinguishing anomalous from behaviour. These models learn the typical time-frequency patterns of the signal during normal operating conditions and can then identify deviations or unusual patterns that may indicate a potential fault or anomaly in the system. In [5], the STFT is used to convert audio data from the MIMII dataset to the frequency spectrum in intervals of 0.1 seconds before using the output of the STFT as the input of an Autoencoder.

In [96], the STFT is used as a 2D input for an Autoencoder whose function is to diagnose faults in roller bearing through vibration data, more specifically using the Case Western University Bearing Dataset [46]. The STFT was favoured over the Wavelet Transform because it demands less prior knowledge compared to the manual selection of a wavelet that best matches the signal's

features, which is required by the Wavelet Transform. The article considers the STFT as a preprocessing step since its output is normalised and only after the latter step is the data fed into the ML model.

### 3.5.3 Time-frequency Techniques

In Section 2.3, state-of-the-art time-frequency analysis methods have been explored, apart from the STFT. This Subsection is dedicated to briefing the use of these techniques in the field of fault detection. [104] mentions the following methods as being extensively used in the research field of interest, more specifically using vibration data:

- Empirical Wavelet Transform
- Empirical Mode Decomposition
- Hilbert-Huang Transform

In [105], the EMD decomposes a vibration signal captured from a machine used in a mining facility into IMFs and faults are detected from the subsequent spectrum analysis created from the IMFs. [106] simulates vibration signals from defective roller bearings and decomposes the signals using the EWT, which allowed for detecting early faults through kurtosis parameters and envelope spectrum of the signals. In [107], a method is developed to improve the performance of the HHT when diagnosing faults in simulated vibration data from rotating machinery.

## 3.6 Wireless Connection

In order to fulfil one of the objectives of this dissertation of developing a wireless sensor module connected to a database for storing its readings. The research fields with the most similarities to this issue are the IoT field and the Wireless Sensor Networks field. IoT was first defined as any physical object that sends data and communicates with a network [108]. Wireless Sensor Networks complement IoT since it is defined as networks of interconnected sensor nodes that wirelessly communicate with each other to collect and exchange data [109].

IoT devices encompass several different use cases, which means many different wireless communication protocols have been used and developed for multiple scenarios. These protocols are distinguished primarily by criteria such as range, data rate or throughput, power consumption and bandwidth [110]. In the dissertation's scenario, the most important features are low range (less than 100 m), the highest data rate possible and the lowest power consumption possible. According to [111], the most used protocols that are considered "low-range" are ZigBee, Bluetooth and WiFi. Table 3.1 compares the listed relevant features. By analysing the table, WiFi is the most suitable candidate for its higher data rate and range. Power consumption was not taken into consideration because the project is not a final implementation, and as such, ease of use was also taken into account, and WiFi fulfilled all the needed criteria.

Table 3.1: IoT communication protocol comparison

| | ZigBee [112] | Bluetooth [113] | WiFi [114] |
|---|---|---|---|
| Frequency Band | 2.4 GHz | 2.4 GHz | 2.4 GHz, 5 GHz |
| Data rate (maximum) | 250 kbps | 3 Mbps | 7 Gbps |
| Range (maximum) | 100 m | 30 m | 100 m |
| Power consumption | 50 mA | 35 mA | 245 mA |

The application layer, according to the Open Systems Interconnection (OSI) model, is the closest layer to the end user. It serves as the interface between the user applications and the underlying network infrastructure. For this project, as previously stated, technologies and protocols used for IoT devices fit the necessary characteristics. Concerning application layer protocols for IoT, the most popular ones are Message Queuing Telemetry Transport (MQTT), Hypertext Transfer Protocol (HTTP), User Datagram Protocol (UDP) and Transmission Control Protocol (TCP) [110]. While several other protocols exist, these were the reviewed ones due to their much higher implementation abundance, which simplifies the research process, as this is not one of the main points of the dissertation. MQTT is the lesser-known protocol, so a small explanation proceeds: it operates on a publish-subscribe messaging paradigm, facilitating efficient communication between devices in the IoT and other distributed systems. The protocol involves two key entities: the MQTT broker and the MQTT clients. The broker acts as an intermediary, receiving messages published by the clients and distributing them to subscribed clients based on predefined topics. MQTT clients can either publish data to specific topics or subscribe to topics of interest. When a client publishes a message to a topic, the broker receives it and forwards it to all subscribed clients interested in that topic. This lightweight and efficient approach minimises network overhead and conserves power, making MQTT ideal for IoT applications with limited resources and intermittent connectivity [7].

## 3.7 Testing Platforms

All of the methods for detecting and diagnosing faults in machinery previously introduced require a platform to validate them. While a significant portion of the articles in the literature rely on data gathered from external sources, such as public datasets (e.g., [46]), some researchers construct their own customised testbeds tailored to the specific requirements of their proposed methods. The majority of articles that employ their own testbeds have very similar characteristics, with the main difference being what component is being monitored, mostly consisting of either bearings or gearboxes. [49] encapsulates the trends of bearing testing equipment and procedures, as PRONOSTIA was developed as a platform for testing PdM algorithms for the IEEE PHM 2012 Prognostic Challenge for the 2012 Prognostics and Health Management Conference [8]. The testbed, displayed in Figure 3.6 contains an AC motor, a gearbox, a bearing, accelerometers, a temperature sensor, a data acquisition card as well other components to couple the mechanical elements.

[7] https://mqtt.org/

[8] https://phmsociety.org/first-european-conference-of-the-prognostics-and-health-management-society-2012/

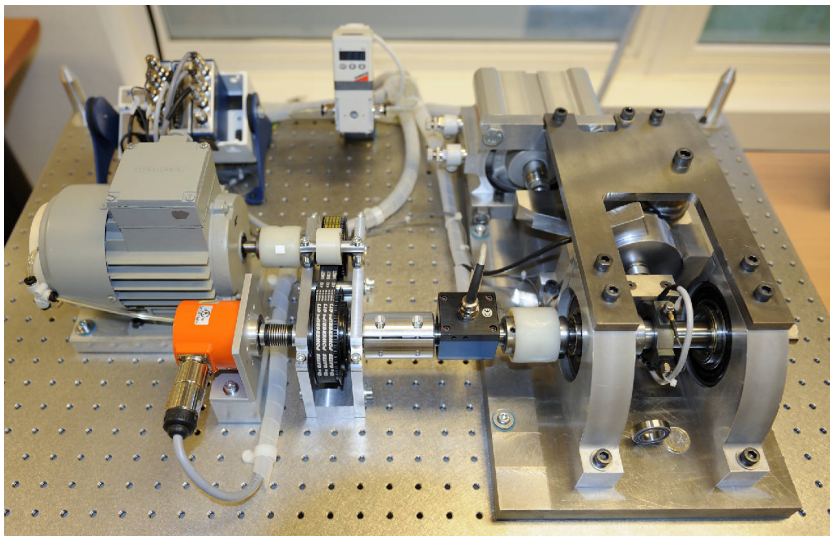


Figure 3.6: PRONOSTIA testbed from [49]

In the PRONOSTIA, the AC motor has a power of 250 W and a rated speed of 2830 rpm. The gearbox converts the rated speed into less than 2000 rpm while maintaining the rated torque. The setup allows the radial force exerted on the bearing to reach its dynamic load limit of 4000 N. The rotational speed is sampled at 100 Hz, the bearing's vibration at 25.6 kHz and the bearing's external ring temperature at 10 Hz. Two one-axis accelerometers are used and placed over the bearing's outer race, one on the horizontal axis and the other on the vertical axis. The accelerometers' model is not specified in the article, but it can be concluded that its bandwidth reaches 12.8 kHz due to the Nyquist Sampling Theorem. For the temperature measurements, an RTD (Resistance Temperature Detector) is used, yet it is also unknown the exact model.

The PRONOSTIA testbed and subsequent dataset is one of the most popular in the literature, however, some articles either custom testbeds are used or opt for a commercial solution. The most prevalent commercial solution is made by SpectraQuest [115]. Multiple models are available; the basic one, named Machinery Fault Simulator, is displayed in Figure 3.7.

The basic simulator includes a 750 W motor and all parts associated with its control, a reconfigurable shaft mount for different sizes and positions, and vibration isolation mounts, among other features to test different elements, such as roller bearings and gearboxes. This simulator has been successfully utilised in several articles, among them [31], [99] and [36], and is most commonly employed for testing heavy machinery components.

## 3.8 Summary

This chapter delved into the core concepts employed in the relevant literature. The exploration encompassed a wide spectrum, ranging from identifying the most suitable components for PdM to exploring the latest advancements in ML algorithms within the domains of Condition Monitoring and Anomaly Detection. The realm of preprocessing techniques was also examined, which play

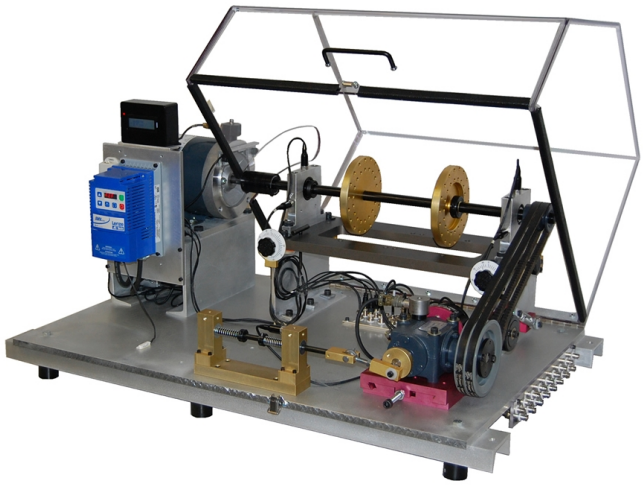


Figure 3.7: SpectraQuest Machinery Fault Simulator [115]

a pivotal role in enhancing model performance, especially in noisy environments. Furthermore, prevalent data types were scrutinised and an in-depth analysis of IoT protocols for wireless communication warranting comprehensive coverage. Finally, the most used testing platforms were analysed to determine the common elements between them.

# Chapter 4

# Methodology

This chapter focuses on detailing the approach and techniques employed to implement the proposed objective. As in the previous Chapter 3, a careful review of which sensors fit the proposed objective while keeping in mind the cost and measurement fidelity. Furthermore, this chapter delves into the communication protocols established for data exchange between the microcontroller and the sensors, as well as between the microcontroller and the database. It offers an overview of the software designed to run on the microcontroller and for the analysis of the captured data. Lastly, it outlines the preprocessing steps applied to the collected data and introduces the models employed.

## 4.1 Prototype Design

This section will dwell on the hardware design choices for the sensor module. More specifically, by analysing different sensor specifications and other details concerning the interface between the microcontroller and the sensors.

### 4.1.1 Vibration Sensors

Vibration can be expressed in three parameters: translational displacement, velocity, and acceleration [44]. Translational acceleration provides information on the rate of change in translational velocity, which describes the speed of displacement. Displacements are larger at lower frequencies and considering that the vibration frequency intervals that better indicate deterioration in the most common machinery fault cause, bearings, are above 150 Hz [116], measuring acceleration is the best method to measure a body's vibration [117].

In order to measure acceleration, displacement transducers or accelerometers can be used. Displacement transducers, such as proximity probes, measure the distance a body has moved in a straight line in relation to the transducer which means that the sensor must be placed in a static position in relation to the ground [44]. This is not possible in the context of the project's objective making accelerometers the only viable option. Accelerometers typically comprise of a suspended mass attached to a spring and damper mechanism housed inside a casing. The accelerometer is

firmly affixed to the object experiencing acceleration and its readings rely on $F_a = M\ddot{x}$, being $F_a$ the force on the body caused by the acceleration, $\ddot{x}$, on the mass, $M$ [117].

The most common accelerometers for vibration measurement in the project context are piezoelectric and MEMS accelerometers [118]. Piezoelectric accelerometers use the piezoelectric effect to output a voltage proportional to the acceleration, thus providing a way to measure it [44]. This category of accelerometer typically provides a measurement resolution of 0.1% of full-scale, with an accuracy of approximately $\pm 1\%$. These devices are available in a range of configurations to accommodate a broad spectrum of measurements and frequencies, spanning from 0.03g full-scale up to 1000g full-scale and from 1 Hz up to 30 kHz, with some reaching up to 90 kHz [118]. Microelectromechanical systems (MEMS) accelerometers are comprised of a micro-machined proof mass that is suspended between two parallel plates from which acceleration is obtained [119]. This type of accelerometer is becoming more popular in the field of condition monitoring as this technology had relatively high internal noise [118], but low-noise MEMS accelerometers are becoming available. The frequency response of these accelerometers are usually much narrower than those of piezoelectric accelerometers, but those of comparable characteristics are much lower in cost.

Taking into consideration the previous statements, the vibration transducers selected were MEMS accelerometers, namely the *ADXL355* [120], using the *EVAL-ADXL355Z* evaluation board in Figure 4.1, and the *MMA8451* [121], using the *Adafruit* breakout board in Figure 4.2. Table 4.1 compares the chosen sensors using the most relevant parameters, particularly sensitivity, noise density and bandwidth.

Table 4.1: Feature comparison between selected accelerometers

| Accelerometer | *MMA8451* | *ADXL355* |
|---|---|---|
| Num. of Axis | 3 | 3 |
| Sensitivity | 4096 counts/g | 256 000 LSB/g |
| Noise Density ($\mu g/\sqrt{Hz}$) | 99 | 22,5 |
| Bandwidth ($Hz$) | 400 | 2100 |
| Price [1] | 7.39 € [122] | 43.46 € [123] |

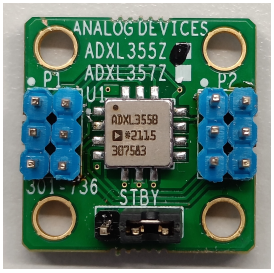



Figure 4.1: *ADXL355* Evaluation Board - *EVAL-ADXL355Z*

[1] At the time of writing

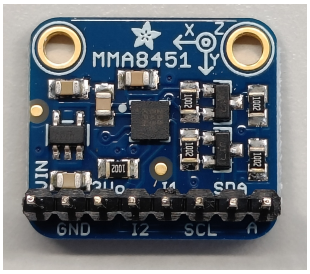



Figure 4.2: *MMA8451* Breakout Board from *Adafruit*

The *Sensivity* parameter in Table 4.1 has different scales for each accelerometer since it was directly transcribed from their respective datasheets. By further analysing the datasheets, the scales express the same unit of measurement, the number of digits used to divide each $g$ (referring to the force per unit mass due to gravity at the Earth's surface, 9.80665 $ms^{-2}$). With this information, it is possible to state that the *ADXL355* is several times more sensitive than the *MMA8451*. Concerning the other parameters in the Table, the *ADXL355* has lower noise density and higher bandwidth. These distinctions are justifiable, given the price differential between the two devices. The *MMA8451* was selected due to the objective of creating a cost-effective sensor module and would serve as a comparison to an accelerometer with more adequate mechanical characteristics, such as the *ADXL355*. For simpler use of the accelerometers, breakout boards were chosen since they remove the need to solder Surface Mount Technology components and the need to build the interaction circuits. However, another advantage that is often overlooked is the fact that the evaluation boards are built with the mechanical structural integrity required for accurate accelerometer readings per the manufacturer's specifications, while a custom solution would have to follow the compulsory guidelines which would delay transducer deployment.

### 4.1.2 Microphones

As seen in Chapter 3, sound analysis is a well-explored method to monitor the condition of mechanical parts that are included in the critical components group. The microphones that are usually chosen appear to be ones that are readily available without much thought given to how the sonic transducer's physical characteristics might affect the measurements and, consequently, the chosen algorithm's results. Nonetheless, it is important to note that this strategy has produced effective results, as remarked previously in the cited articles, which means there is little incentive to switch approaches. As the ultimate goal of this project is the deployment of the prototype in an environment with very high levels of noise (an industrial setting), more thought was put into choosing the right microphones for the situation.

#### 4.1.2.1 Electret vs. MEMS

The most adequate microphone types for this prototype are of the Electret and MEMS varieties due to their relatively small dimensions when compared to other technologies, since Electret mi-

crophones do not exceed 8 mm in diameter and 4 mm in length and MEMS microphones do not have any size measurement over 4 mm. Although Electret microphones are the most popular choice, MEMS microphones are the better option for the use case scenario, due to their compactness, ruggedness, and resistance to environmental factors and lower power consumption. MEMS microphones also posses better noise performance and a higher signal to noise ratio (SNR) in relation to Electret ones [124]. Additionally, it is noteworthy that MEMS microphones can be used as ultrasonic transducers, as some models present its frequency response over the human hearing range of 20 kHz. [125] used MEMS microphones to build an ultrasonic microphone array to inspect concrete for faults using ultrasonic wavefield imaging.

#### 4.1.2.2 Microphone Selection Criteria

With hundreds to thousands of MEMS microphone models to choose from, guidelines need to be established in order to select the one with the most appropriate features for the use case scenario. Two considerations other than environmental resistance have already been determined: noise performance and SNR. One way to create guidelines is to check the specifications of the microphones used in the already reviewed literature. As many of the previously cited articles use the same three datasets, MIMII [58], ToyADMOS [57] and ToyADMOS2 [59], Table 4.2 compares features from each microphone used. The model used in [63] was added to the table since a specialised measurement microphone was chosen while the other articles used general-purpose microphones for video recording usage. The MIMII dataset uses the TAMAGO-03 microphone array but the datasheet was not available, so all values are missing from Table 4.2. However, it is possible to infer that the frequency range is probably up to 20 kHz since the product page [2] lists the product as a device to aid video calls.

A quick examination of Table 4.2 reveals that most information is not available to the consumer, including SNR and noise performance. These two parameters tend to be only displayed in the datasheet for the sound transducer itself, for which consumer microphones do not list, so it is not possible to know these unless directly contacting the manufacturer. This is the justification for the prior statement relating to the lack of criteria for selecting microphones, given that certain characteristics that greatly affect measurements are not available and those microphones are still used. The Shure SM-11 and the GRAS 40PH are the ones with the most available information, so a deeper analysis can be performed.

Regarding sensitivity, the GRAS 40PH has much higher sensitivity, since the higher the output voltage for the same pressure, the higher the sensitivity and the GRAS has an output voltage over 150 times higher than the SM-11. Microphone output impedance affects the amount of noise and signal quality that the microphone's output endures and the lower the output impedance, the lower the noise and higher the signal quality. Frequency response details how the output varies across the frequency spectrum. The relative superiority of the GRAS 40PH over the SM-11 is to be expected, because the first one was developed to be a measurement microphone, meaning its goal

[2] https://www.sifi.co.jp/en/case/hylable/

is to capture sound in the most accurate way, while the SM-11's objective is to capture the human voice in the clearest way, minimising unwanted environmental noise.

Table 4.2: Feature comparison between microphones used in cited datasets

| Dataset | ToyADMOS | ToyADMOS2 | MIMII | Milling tool |
|---|---|---|---|---|
| Microphone Model | Shure SM11 | EM-700 | TAMAGO-03 | GRAS 40PH |
| Sensitivity [3] | 0.6 | Not available | Not available | 100 |
| Output Impedance (Ω) | 150 | 400 | Not available | >50 |
| Frequency Range (Hz) | 50 - 15 000 | 20 - 20 000 | Not available | 50 - 20 000 |
| Frequency Response | Not available | Not available | Not available | Figure 4.3 |
| SNR | Not available | Not available | Not available | Not available |
| Noise Performance | Not available | Not available | Not available | Not available |

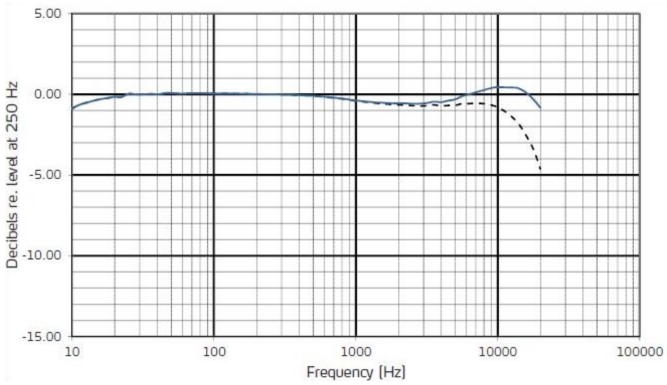



Figure 4.3: GRAS 40PH Microphone Frequency Response from [126]

The final overview for the selection criteria is that these should include the frequency response, sensitivity, frequency range and SNR, if available. These features describe how the microphone's characteristics influence readings and the considerations that should be taken in order to retrieve measurements with the highest amount of fidelity.

#### 4.1.2.3 Chosen Microphones

One of the objectives of this dissertation is to make use of cost-effective sensors in the final solution. With this goal in consideration, the SPH0645LM4H digital microphone [127] (using a breakout board from *Adafruit* [4]) was selected. Even though certain criteria were delineated in SubSubSection 4.1.2.2, the selection of this microphone was predominantly driven by its cost of 6,46 € [128], at the time of writing. Another significant factor contributing to this decision was the absence of established criteria in the literature for selecting a suitable microphone while consistently achieving accuracies surpassing 90% in the final models. This gap in guidance prompted the exploration of potentially employing a microphone originally designed for smartphone integration in the context of detecting anomalies within mechanical systems.

[3] (mV/Pa @ 1 kHz)
[4] https://www.adafruit.com/product/3421

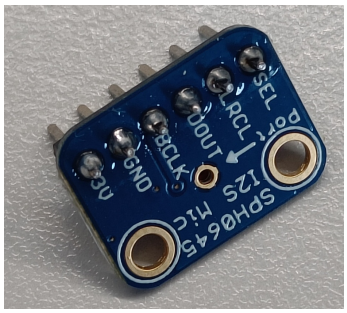



Figure 4.4: *SPH0645LM4H* Microphone Breakout Board from *Adafruit*

The SPH0645LM4H uses the Inter-IC Sound (I2S) communication protocol. Figure 4.5 displays the microphone's frequency response plotted up to 10 kHz. The labels on the graph refer to the frequency of the microphone's internal clock, which is set by the BCLCK frequency of the I2S protocol that the controller device provides, according to the datasheet [127]. The standard frequency is 3.072 MHz, meaning the frequency response is characterised by the light blue line, according to the label. Table 4.3 exhibits the other main characteristics of the microphone.

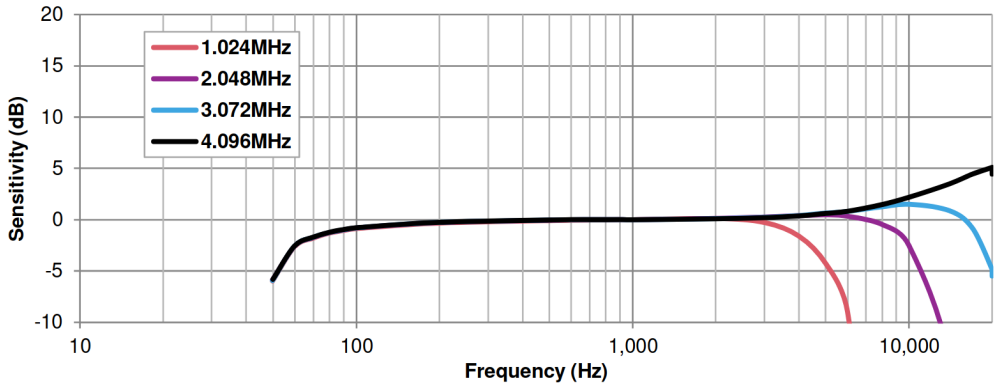



Figure 4.5: *SPH0645LM4H* Microphone Frequency Response, from [127]

Table 4.3: SPH0645LM4H's Main Features

| Microphone | SPH0645LM4H |
|---|---|
| Sensitivity (dBFS @ 1 kHz) | - 25 |
| Bandwidth (Hz) | 50 - 15 000 |
| SNR (94 dB SPL @ 1 kHz) (dB(A)) | 65 |

As previously mentioned, especially in Subsection 3.2.3, some articles have found better results regarding anomaly detection when analysing vibration frequency at higher frequency ranges, which is the basis for Acoustic Emission. However, all of the cited articles that use sound analysis, tend to adhere to utilising consumer microphones, made for recording human perceptible sounds, up to 20 kHz. This might be due to the relatively higher prices of ultrasonic microphones

and limited available selection. Moreover, the name "ultrasonic microphone" is hardly used, with the term "ultrasonic transducer" being much more popular. Furthermore, "ultrasonic transducer" is usually associated with devices for measuring distance between the sensor and the object, e.g. the ever-popular HC-SR04 [5]. These sensors could be used for detecting ultrasonic waves created by faulty parts, but their frequency response is only typically charted between a small interval of frequencies, as with the CUSA-R75 whose frequency response is plotted between 37 and 43 kHz. This fact is detrimental to the experiment since the exact frequency of the faulty part sound signal is not known and a relatively small interval of captured signal might not acquire the sought-after signal. It has also

At the time of writing, MEMS technologies have enabled cheaper microphones to be mass-produced, with very few of the models having a charted frequency response above 20 kHz. However, these do exist and are available at a low enough price to be included in the project's objectives, which include using low-cost sensors. The selected ultrasonic MEMS microphone was the SPU0410LR5H-QB [129]. Figure 4.6 shows the selected microphone in a breakout board since MEMS components are very small when compared to "regular" components. The SPU0410LR5H-QB's length is 3.76 mm, which makes it very difficult to solder and a breakout board is needed for easier prototyping. Another important feature to note is that the SPU0410LR5H-QB's output is an analog voltage, with a maximum peak-to-peak voltage of, so an amplifier circuit is needed to fully utilise the Analog-to-Digital Converter's (ADC) resolution capabilities.

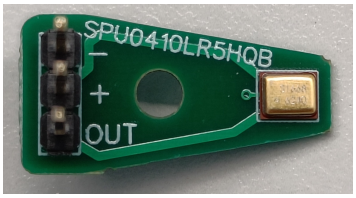



Figure 4.6: SPU0410LR5H-QB Microphone Breakout Board

Table 4.4 lists the main characteristics of the ultrasonic microphone. These were not extensively selected due to the very low amount of MEMS ultrasonic microphone models available. In Figure 4.7, the frequency response is plotted, from 10 to 80 kHz and a cursory look at the chart reveals that at these frequencies, the output is not linear throughout the interval and this fact must be taken into account when amplifying the analog output voltage and when processing the data.

[5] https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf

Table 4.4: SPU0410LR5H-QB's Main Features

| Microphone | SPU0410LR5H-QB |
|---|---|
| Sensitivity (mV/Pa @ 1 kHz) | 12.6 |
| Output Impedance (Ω) | 400 |
| Frequency range (Hz) | 100 - 80 000 |
| SNR (94 dB SPL @ 1 kHz) (dB(A)) | 63 |
| Price [6] | 14.76 € [130] |

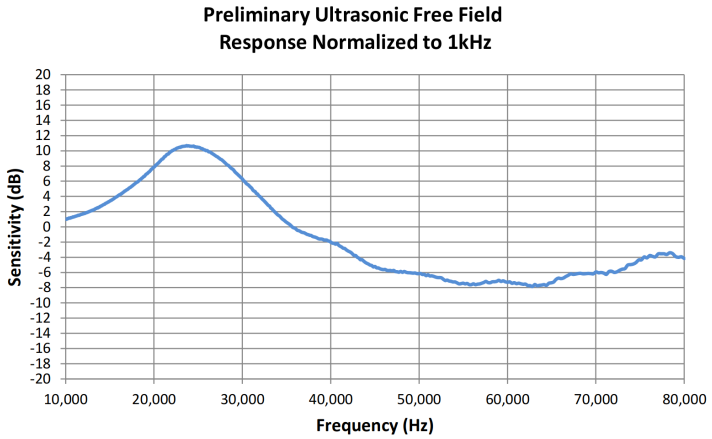



Figure 4.7: SPU0410LR5H-QB Microphone Frequency Response over 10 kHz [129]

#### 4.1.2.4 Analog Signal Amplification

Analog signal amplification is a crucial process in the context of MEMS microphones, which are highly sensitive transducers used to convert sound waves into electrical signals. Due to the small size of MEMS microphones, the generated electrical signals are typically very weak. To ensure accurate and reliable signal detection, the analog signals from the microphone need to be amplified before further processing.

The SPU0410LR5H-QB microphone outputs an analog voltage, so two different steps are required before having audio data capable of being analysed by an anomaly detection algorithm: amplifying the output and sound digitisation. In this Subsection, the first step's proposal is delineated.

The first step in designing an amplifier circuit is to check the requirements. Further ahead will be made clear the choice of ADC, which has a voltage range between 0 and 3.3 volts. The output of a microphone is a symmetrical sinusoidal signal centred around 0 V, meaning the output must be given an offset of half the maximum input value of the ADC in order to take advantage of the full scale of the ADC. The next step is to check the microphone's characteristics, available in its datasheet [129]. The DC output with a supply voltage of 1.5 V is typically 0.73 V. The output

[6] At the time of writing

sensitivity at a sound pressure level of 94 dB at 1 kHz has a maximum of -35 dBV/Pa which is equivalent to 17 mV/Pa. Since the selected microcontroller ADC's maximum input is 3.3 V and considering the 1.65 V offset, the gain the amplifier needs to provide is 97. It is necessary to point out that the output is not constant, as remarked by the frequency response graph, present in Figure 4.7, but the gain must take into account the maximum output so as not to oversaturate the audio, which causes loss of information, also known as *clipping*. Amplifier circuits for MEMS microphones are not generally not publicly accessible, but in [131], an open-source acoustic device for monitoring biodiversity is detailed, including schematics for an amplifier circuit. One of the objectives of the device is to detect ultrasonic sounds produced by bats, so it uses a previous iteration of the model of the microphone chosen for this dissertation, the SPM0408LE5H-TB.

The final amplifier circuit design for this project is presented in Figure 4.8 and follows the same configuration of two inverting operational amplifier circuits, using the OPA350 [132], as the aforementioned one present in [131]. The OPA350 was selected due to its noise of 5 nV/$\sqrt{Hz}$. Although the use of a single operational amplifier appears to be intuitive, it was observed that that configuration would introduce an amount of electrical noise in the circuit's output that would completely obscure the sound signal's voltage.

A further inspection of the frequency response chart in Figure 4.7 reveals that the sensitivity is very uneven over the charted frequency values, and a peak exists around 25 kHz. Since the goal of using this specific microphone is to capture ultrasonic waves, a passive band-pass filter was made to ensure the readings are not overly influenced by signals in the 10 to 40 kHz range. The region of interest and of lower variation is between 45 and 80 kHz, which was the band chosen for the signal to pass through. A passive band-pass filter can be made using a passive low-pass filter and a passive high-pass filter directly connected to each other. The filter's resistance and capacitance were calculated using Equation 4.1, where $f_c$ represents cut-off frequency. This calculation was performed twice: once for the low-pass component and once for the high-pass component of the filter. The filter is connected to the output of the microphone, as seen in Figure 4.8, and its components are identified by either *bp_h* (band-pass high) or by *bp_l* (band-pass low).

$$f_c = \frac{1}{2\pi RC} \tag{4.1}$$

Table 4.5 contains all the values of the passive components used in the amplifier circuit in Figure 4.8. The SPU0410LR5H-QB's datasheet mentions that capacitors placed close to the microphone should not contain Class 2 dielectrics, which are a type of dielectric used in certain ceramic capacitors. This recommendation is due to the Piezoelectric Effect in this type of capacitor, as current passes through the capacitors, these expand and contract, which causes vibration and affects the signal captured by the microphone [133]. The designed circuit will be built on a breadboard, so this recommendation was ignored since the placing of the components is much more unstable and susceptible to other sources of vibration than the Piezoelectric Effect affecting the capacitors.

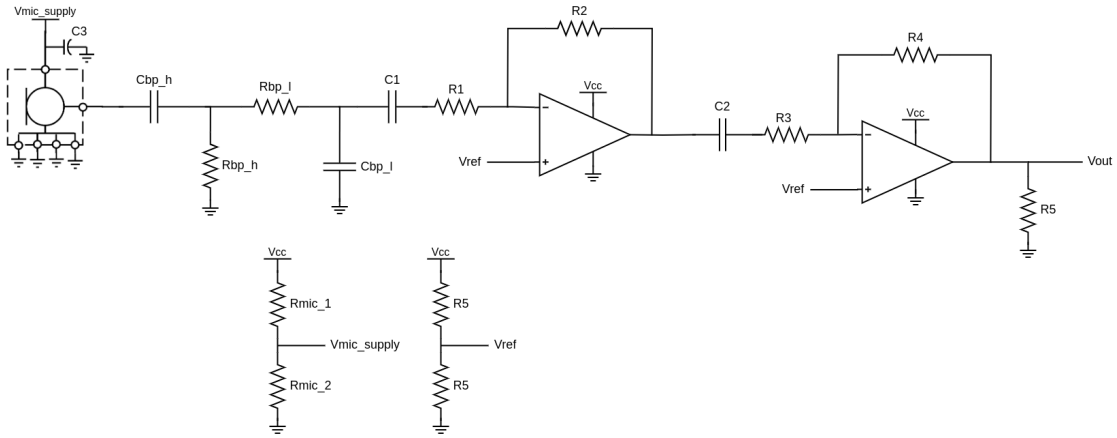



Figure 4.8: Microphone Amplifier Circuit

Table 4.5: Amplifier Circuit Passive Component Values

| Passive Component | Value |
|---|---|
| Rmic_1 | 12 k Ω |
| Rmic_2 | 10 k Ω |
| Rbp_h | 1k8 Ω |
| Rbp_l | 2k2 Ω |
| R1 | 1 k Ω |
| R2 | 10 k Ω |
| R3 | 1 k Ω |
| R4 | 10 k Ω |
| R5 | 22 k Ω |
| Cbp_h | 1 nF |
| Cbp_l | 2.2 nF |
| C1 | 100 nF |
| C2 | 100 nF |
| C3 | 100 $\mu$F |

### 4.1.3 Temperature Sensors

Certain applications necessitate the use of non-contact temperature sensors for measuring component temperatures. One such scenario arises when assessing the inner raceway of a bearing. Utilising a contact sensor in this context can disrupt the system's dynamics, leading to undesirable inaccuracies that fail to represent the system's true performance accurately. IR thermometers are used as non-contact sensors since they possess the ability to detect the IR radiation from objects, taking into account the object's emissivity, as seen in Subsection 3.2.4.

Keeping in line with one of the goals of this dissertation, an IR thermometer was chosen, the MLX90614 [134], displayed in Figure 4.9. Table 4.6 presents the key specifications of the thermometer. The "Sensor Temperature Range" specifies the ambient temperature range within which the sensor is guaranteed by the manufacturer to operate within its stated specifications. The chosen breakout board is sourced from *seeed studio*[7], selected primarily for its ready availability and lower cost.

Table 4.6: MLX90614 Specifications [134]

| | |
|---|---|
| Sensor Temperature Range | -40 to +125 ºC |
| Object Measurement Range | -70 to +380 ºC |
| Accuracy | ± 0.5 ºC |
| Resolution | 0.02 ºC |
| Field of View | 35º |
| Refresh Rate | 0.5 to 64 Hz |
| Digital Communication Protocol | I2C |
| Price [8] | 25.14 € [135] |

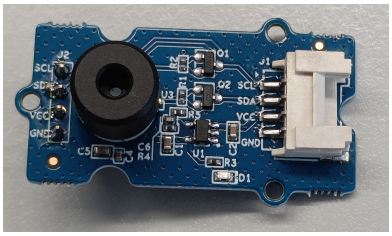



Figure 4.9: *MLX90614* IR Thermometer

### 4.1.4 Microcontroller

According to the chosen sensors, the prototype's microcontroller needs to fulfil certain requirements, namely:

- Support for Inter-Integrated Circuit (I2C)
- Support for Serial Peripheral Interface (SPI)
- A CPU capable of processing data with sampling rates over 1 MHz
- A high-resolution integrated ADC is not mandatory but welcomed

The selected microcontroller is the Teensy 4.1 [9], displayed in Figure 4.10 from PJRC. It has an ARM Cortex-M7 CPU [136] with a maximum clock speed of 600 MHz, 2.14 DMIPS/MHz and

[7] https://www.seeedstudio.com/Grove-Thermal-Imaging-Camera-MLX90614-DCC-IR-Array-with-35-FOV-p-4657.html
[8] At the time of writing
[9] https://www.pjrc.com/store/teensy41.html

includes a Digital Signal Processing hardware accelerator. It also possesses three units capable of communicating over I2C and two others capable over SPI, although either protocol is able to communicate using only one bus for multiple devices [137]. The last feature worthy to point out is the fact that the Arduino framework [10] is available, which greatly simplifies the programming of the microcontroller as the Arduino libraries are available.

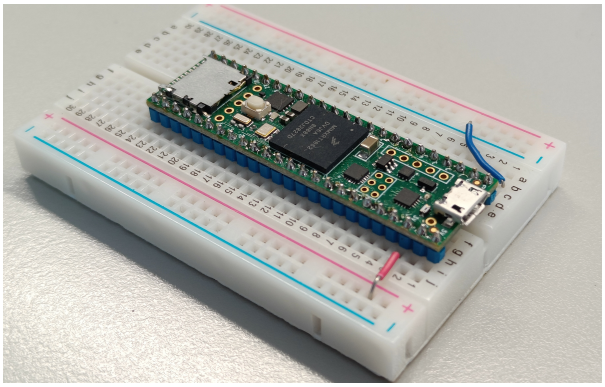


Figure 4.10: Teensy 4.1 Microcontroller

### 4.1.5 Voltage Incompatibility

Some of the chosen sensors, such as the MLX90614, operate at 5 V, whereas the Teensy 4.1 microcontroller is designed to interface with 3.3 V compatible devices. Attempting to directly connect these sensors to the Teensy without proper voltage level translation may result in potential issues such as voltage mismatch and could lead to malfunction or damage to both the sensors and the microcontroller.

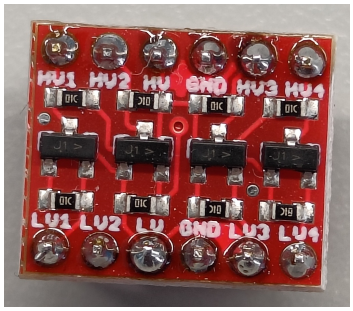



Figure 4.11: *Sparkfun* Bi-directional Logic Level Converter

Given that the digital communication protocols for the mentioned sensors are all based on I2C, this must be taken into account for choosing a method for converting the 5 V signals into 3.3 V and vice-versa. While alternative methods may come to mind, utilising a dedicated logic level converter is the less complex and more practical solution. These devices are specifically designed to ensure reliable voltage translation, minimising the risk of communication errors or

[10] https://www.arduino.cc/

damage to the components. The *SparkFun* Bi-directional Logic Level Converter [138], in Figure 4.11 provides these advantages while also being priced at 2,74€.

### 4.1.6 Analog-to-Digital Conversion

Considering the choice of the SPU0410LR5H-QB ultrasonic microphone, it becomes necessary to utilize an Analog-to-Digital Converter (ADC) to convert the analog output signal from the amplifier circuit into a digital format for analysis alongside other sensors. The Teensy 4.1 microcontroller is equipped with a 12-bit ADC that can achieve a sampling rate of 1 million samples per second, as per its datasheet.

While external ADCs with higher resolutions (e.g., 16-bit and 20-bit) and faster sampling rates are available, the decision to opt for the integrated Teensy 4.1 ADC was made due to the use of a 12-bit ADC in [131], and the incorporation of an external ADC would introduce additional implementation complexities.

It's worth noting that while the Teensy 4.1 is supported by the Arduino Framework, the standard library does not encompass all of the microcontroller's functionalities. Therefore, the integration of an external library is necessary. Fortunately, an open-source library recommended directly by the manufacturer, known as the ADC library [11], proves to be a valuable solution in this context.

### 4.1.7 Hardware Connections

Figure 4.12 contains a diagram of the hardware connections between the Teensy 4.1 and the selected sensors. It is worth noting that the diagram does not represent the physical location of the pins on the microcontroller; the labels are meant to aid in analysing the connections.

### 4.1.8 Local Processing

By selecting the Teensy 4.1 microcontroller, it is possible to take advantage of the ARM Cortex-M7 processor's floating point unit, which enables the use of the *CMSIS-DSP* [12] library from ARM Software. This library is an optimized compute library for embedded systems which takes advantage of the available hardware and improves speed and efficiency for certain algorithms, such as complex and real FFT, matrix functions (e.g. inverse matrix and matrix multiplication), quaternion functions and statistical functions (e.g. root mean square). It also has support for SVM run-time optimization, if the training stage is performed using *scikit-learn*.

Other libraries were also considered, namely the *ArduinoFFT* [13] and the *FFTW* [139] library. The first one due to its ease of use, wide range of examples and documentation and the fact that it is meant for the Arduino Framework which makes it more intuitive while using this framework for the rest of the main program. The second one was considered by virtue of its efficient performance [14], but its implementation is not optimized for embedded CPUs and the *CMSIS-DSP* FFT set of

[11] https://github.com/pedvide/ADC
[12] https://github.com/ARM-software/CMSIS-DSP
[13] https://github.com/kosme/arduinoFFT
[14] https://www.fftw.org/speed/

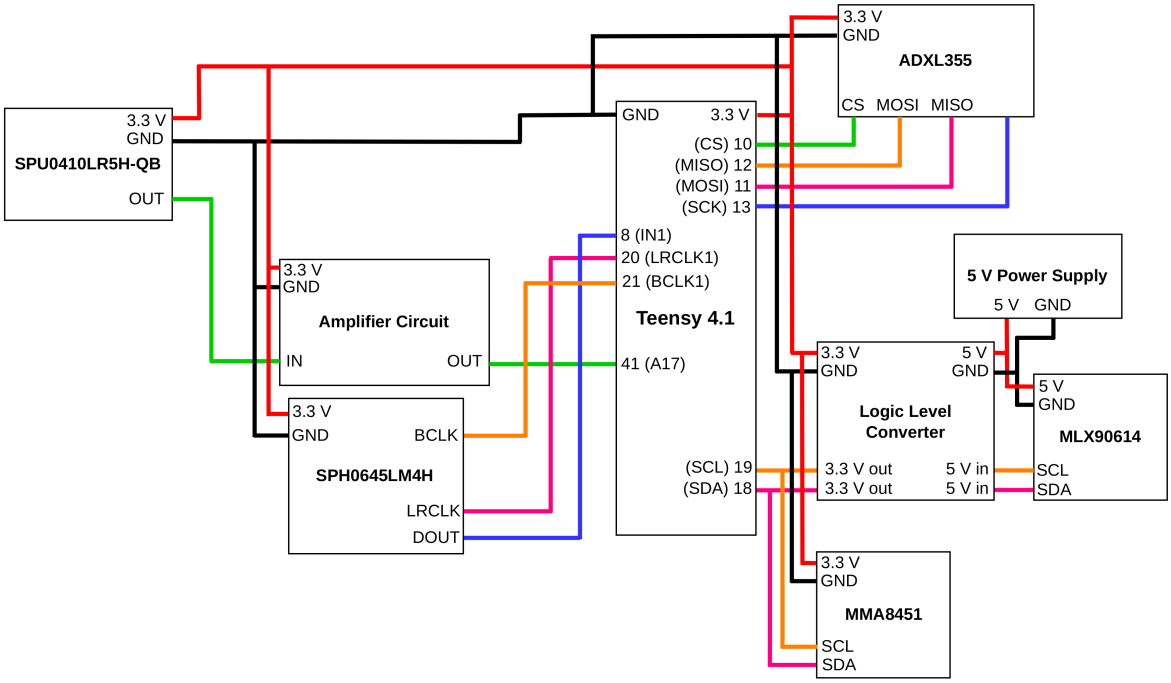



Figure 4.12: Sensor Module Hardware Connection Diagram - Without ESP-12E

functions uses the Cortex-M7's capabilities to their most efficient operation.

### 4.1.9 WiFi Connection

One of the primary goals of this dissertation is to establish a wireless sensor module capable of connecting to a database for storing its readings. The chosen physical layer protocol for this purpose is WiFi, owing to its suitable range and data transfer rate, which align well with the project's environmental conditions and use case requirements. The selected microcontroller, the Teensy 4.1, does not have WiFi capabilities, so a separate microcontroller is needed in order to transmit data wirelessly. One of the most popular WiFi chipsets available is the ESP8266 [140], and the selected microcontroller which contains the chipset was the ESP-12E [141] (in Figure 4.13) due to its immediate availability. At the time of writing, newer and more capable models have been launched, e.g. the ESP32-C6 [142], but the ESP-12E is supported by the Arduino Framework, which simplifies programming and enables access to the existing libraries made for that framework. It is also noteworthy the amount of documentation which is available for this microcontroller, which simplifies troubleshooting and implementation of programs.

The ESP-12E is a microcontroller, however, it is not suitable for the project's requirements, since it lacks processing power, enough SPI and I2C connections, ADC speed and accuracy and digital ports for all the selected sensors. So, the best possible combination of hardware for the project's needs is to merge both microcontrollers, using the Teensy 4.1's processing and I/O characteristics and the ESP-12E WiFi connectivity. A way to communicate between the Teensy 4.1 and the ESP-12E is needed, for which three possible approaches exist: UART, I2C or SPI, as both

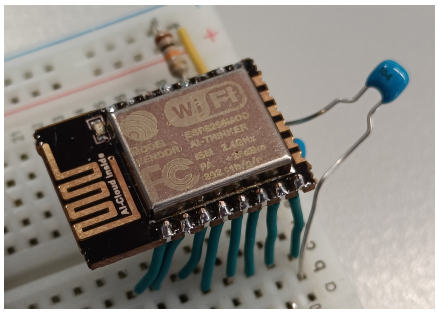


Figure 4.13: ESP-12E Microcontroller

microcontrollers share these communication methods. Table 4.7 compares the most relevant characteristics in relation to each other, namely, speed, ease of use and the number of wires needed between the microcontroller boards.

Table 4.7: Serial Communication Protocol Comparison

| Protocol | UART | I2C | SPI |
|---|---|---|---|
| Relative Speed | Slowest | Intermediate | Fastest |
| Relative Ease of Use | Easiest | Hardest | Intermediate |
| Number of Wires | 2 | 2 | 4 |

After analysing Table 4.7, the most suitable protocol is either UART, due to its ease of implementation and minimal connection usage, or SPI, which offers relatively higher data transfer speeds. In this context, UART has been selected as the preferred protocol, considering that this project primarily serves as a proof of concept and does not demand the utmost performance. Thus, opting for simpler implementation over faster communication is a valid and pragmatic decision.

The ESP-12E lacks a straightforward means of connection to a computer in order to be programmed, so external hardware is needed. It is also important to note that the microcontroller has a preloaded program which enables simple actions via a Universal Asynchronous Receiver/Transmitter (UART) connection using AT Commands to control the ESP8266 chipset, e.g. connecting to an access point with *AT+CWJAP* or sending data with *AT+CIPSENDEX*. This method of interaction is more restrictive and less user-friendly than simply programming it, which can be achieved via its UART pins and setting some pins to *HIGH* or *LOW* to put it in *programming mode*. The *FT232RL* [143], in Figure 4.14 is a Universal Serial Bus (USB) to Serial adapter which enables the programming of the ESP-12E using a computer with a USB port.

Figures 4.15 and 4.16 display the hardware connections in order to program the ESP-12E and to connect to the Teensy 4.1, respectively. It is important to point out that although these connection schemes function for interacting with the ESP-12E, the connection is unstable and requires extra capacitors to negate instability in the power supply.

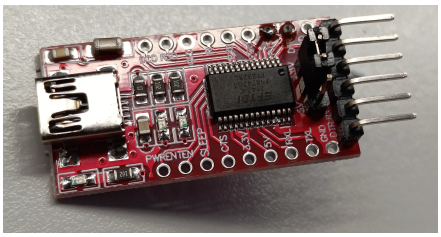


Figure 4.14: FT232RL USB to Serial

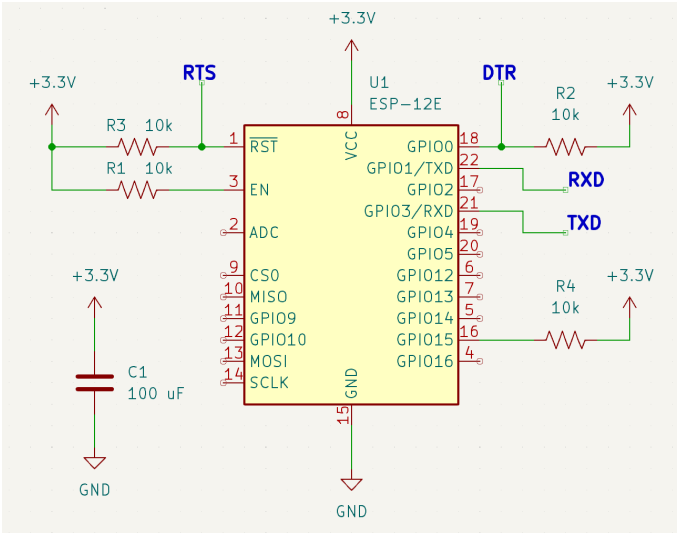



Figure 4.15: ESP-12E Programming Mode connection diagram

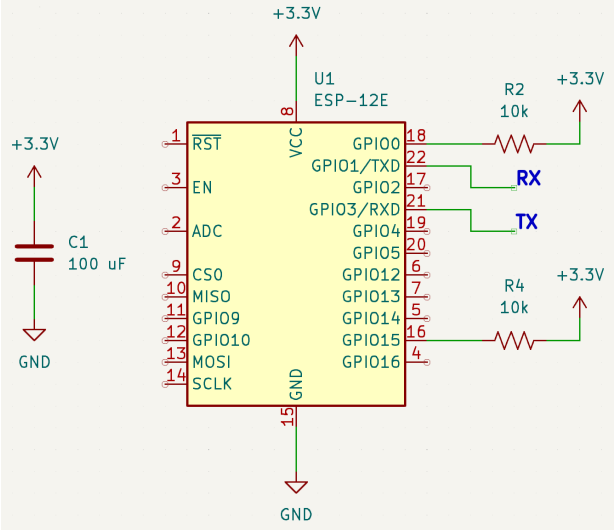



Figure 4.16: ESP-12E Serial connection with the Teensy 4.1

The Teensy 4.1 is equipped with eight serial ports, and to transmit data to the ESP-12E through one of these ports, the function *Serial1.print* is employed. To verify the successful transmission of data, the ESP-12E can utilise the *Serial.read* function. For error checking and to ensure all

information has been sent and correctly received, an application layer must be created, which has not been accomplished, but small tests allowed for verifying that this method is valid.

### 4.1.10 Overview

Figure 4.17 contains a simplified diagram with the selected sensors and the connection method from the microcontroller to the sensors. The sample rate of the sensors is also displayed as it is the feature with the most impact on the program's structure due to the time restrictions it imposes.

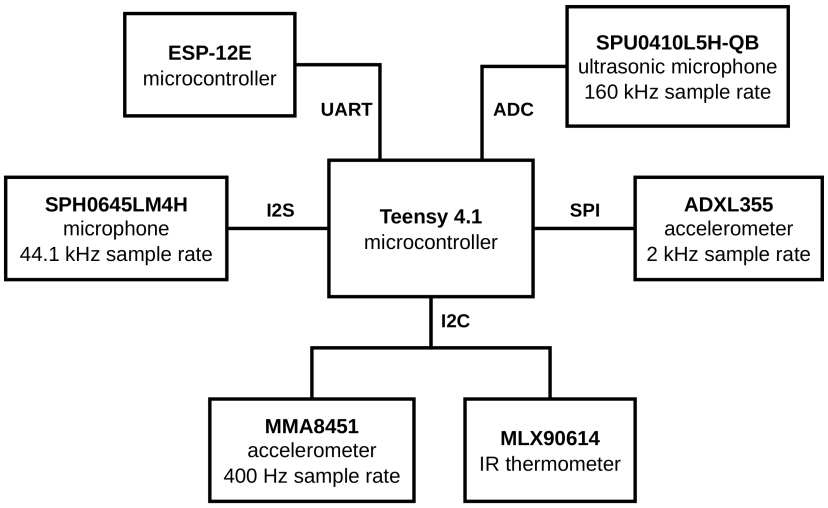



Figure 4.17: Sensor Module Hardware Overview

## 4.2 Database

As with any sensor system, a countless amount of data is generated, especially considering the accelerometers and ultrasonic microphones for which data must be captured at least 160 thousand times each second, a way to store these measurements must be thoroughly considered. The Teensy 4.1 has support for a micro SD card [15] for which storage at the time of writing can reach up to 1.5 TB [16], but this would not be a practical solution, since it would require an operator to retrieve the data from the SD card manually. The most practical solution for dealing with data storage management is to employ a database, analogous to how IoT devices store data. Using a database would also enable interconnectivity between similar sensor modules or other hardware to run the developed statistical and ML algorithms.

Time-series databases are a specialised category of databases designed specifically for efficiently storing, managing, and analysing time-stamped data. Time series data, commonly generated by IoT devices, sensors, or monitoring systems, consists of measurements or events that occur over time. Unlike traditional relational databases such as *PostgreSQL*, which can handle

[15] https://www.sdcard.org/
[16] https://www.micron.com/products/memory-cards

time-series data to some extent, time-series databases are optimized for storing time-stamped data that of high-frequency ingestion.

The most popular time-series databases, at the time of writing, are: *InfluxDB*, *Prometheus* and *DolphinDB* [144]. The chosen one was, ultimately, *InfluxDB* due to being the most popular one, meaning it is the one with the most documentation and readily available examples. The cloud-based servers can be used for free, which alleviates the amount of data storage needed for the present iteration of this project's prototype. Several different web client APIs are also available, including one for the Arduino Framework which uses either the ESP32 or ESP8266 WiFi chipsets which can be used since the ESP8266 was the chosen WiFi connection hardware method chosen for the prototype.

While MQTT is a widely used and efficient protocol for communication between multiple devices in an IoT (Internet of Things) system, it may not be the best fit for a project that involves only one device. MQTT is designed to facilitate communication between multiple devices, enabling efficient data exchange and real-time updates in a network of interconnected nodes. In a scenario where there is only one device, using MQTT may introduce unnecessary complexity and overhead. Alternative communication protocols, such as HTTP or simple serial communication, may be more straightforward and suitable for a single-device project. InfluxDB has an Arduino Client API for ESP32 and ESP8266 which allows for a simpler implementation route.

## 4.3 Custom Testbed

Before testing the final prototype in an industrial setting, a controlled environment is required to fully test the prototype to ensure no unexpected behaviours occur in the final test. The process of guaranteeing the prototype's viability goes through a few steps that need to be checked:

- Sensors output correct measurements according to their datasheets;
- Measurements are sent to the database;
- Communication error handling;
- Prototype's soldered connections do not break due to excessive vibration;
- Sensor position in relation to the target component;

The previous steps only refer to the prototype's physical integrity and reliability, nonetheless, using a testbed also brings another obvious advantage: being able to test the anomaly detection algorithm with a controlled measurement noise level. Noise in this context refers to spurious vibrations and sounds that, in an industrial setting, might be caused by several different sources. These are not limited to environmental causes, e.g. other machines operating, but also by the target machine, as is the case with structural resonances. This noise source should be taken into account, as exemplified in [145], where a simulation model of a bearing was created, and structural resonance is part of the equation that characterises the degradation process.

The first and most logical step for knowing which hardware to use in order to create a testbed suitable for the dissertation's objective is to compare testbeds in articles with similar goals of condition monitoring and anomaly detection, which have already been explored in Section 3.7.

The testbeds previously explored in Section 3.7 are noticeably guided towards emulating heavy machinery, however, the goal of this dissertation consists of detecting anomalies in an AGV and accurately replicating the generic setup in the literature does not represent the critical components of an AGV due to the differences in size and power of the elements, such as the motors and shaft diameters. Most characteristics of the analysed testbeds can be adapted to fit the objective, namely the structure elements of a motor and gearbox, a mechanical coupler, a shaft, a bearing and a mechanical load. This structure can be used to represent the traction system of the AGVas it comprises motors, gearboxes, shafts, bearings and wheels. Environmental noise is a big concern in the ultimate scenario, and this factor should be taken into account while testing the prototype, however, it is mostly ignored in the literature. In [82], a Bluetooth device is created, and sound is used to identify faults in rolling bearings and to simulate real conditions, a speaker playing prerecorded sounds from a similar environment as the planned deployment space is added to the testing procedures to evaluate the prototype's performance with and without extra noise. For this dissertation, the same concept can be used to add noise to the overall system, however, parasitic vibration is more difficult to simulate due to the need for costly equipment, such as a controllable vibrating table.

To validate that the final sensor module is capable of detecting anomalous behaviour in the face of noise and different scenarios, three different testbeds were built. In order to more accurately simulate the AGV, components that are within the required scale were selected. The chosen basis components were:

- A 12 V DC motor
- A mechanical decoupler connecting the motor shaft to the main shaft
- Two ball bearings for securing the shaft onto the platform
- An aluminium shaft
- A central ball bearing for testing purposes
- A wooden base to secure all the components

All testbeds use the same fixating ball bearing setup while varying the motor and main shaft length, in Figures 4.18 to 4.20. The first testbed, in Figure 4.18, is comprised of a 1000 rpm DC motor and a 40 cm shaft, the second testbed, in Figure 4.19, includes a 1000 rpm DC motor and a 10 cm shaft and the third testbed, Figure 4.20, a 2640 rpm DC motor and a 10 cm shaft. The black pieces visible in all testbeds were 3D printed and served as a support for the motor to rest on, otherwise, the motor would hang and would only be supported by the decoupler.

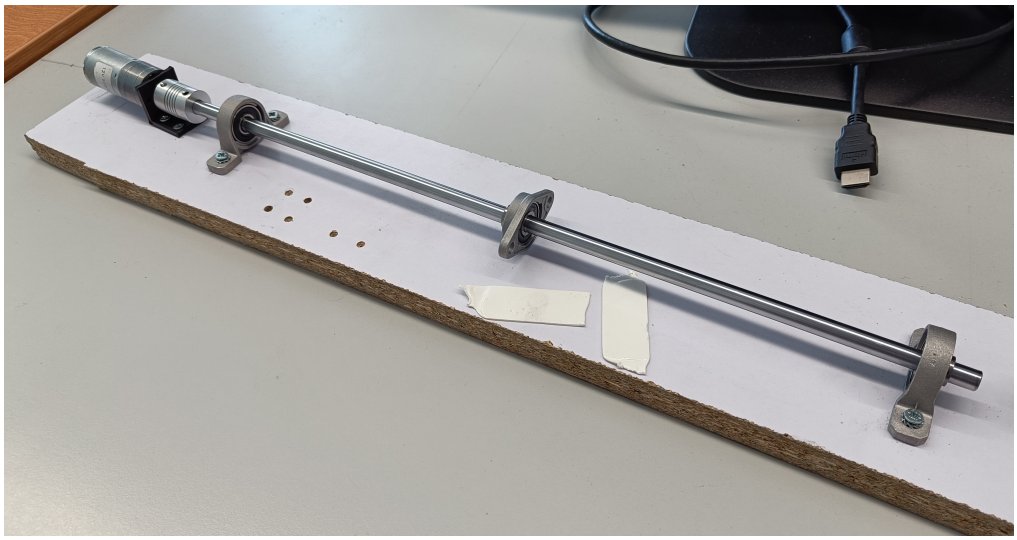


Figure 4.18: Testbed 1 without any sensors

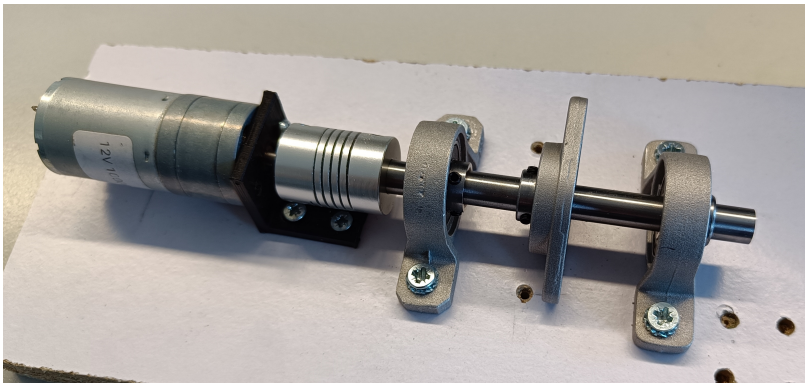


Figure 4.19: Testbed 2 without any sensors

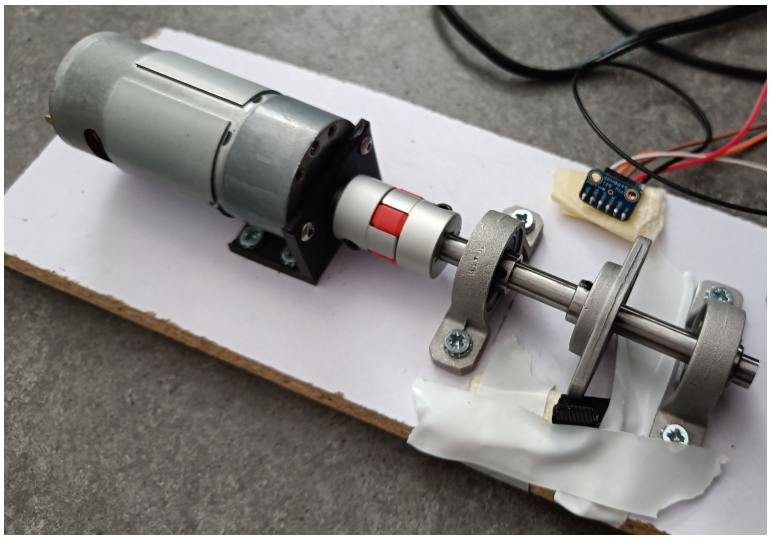


Figure 4.20: Testbed 3 with the *SPH0645LM4H* microphone visible

On the reviewed testbeds, the ones using accelerometers usually place them either on top of the bearing support, on top of the motor or gearbox under testing or on the main shaft. However, in this case, due to the physical dimensions of the AGV and, thus, of the testbed, it is not possible to fixate the accelerometers on any of these locations. Since the wooden base is attached to the testbed, the vibrations of the setup also propagate to the base, making it a viable option to place the sensors. This is analogous to placing the sensors in the AGV since space inside is limited, and the only feasible option is to put the sensors as close as possible to the critical components. The placement of the temperature sensors and microphones are not as carefully placed as these can take measurements without being physically connected to the object.

### 4.3.1 Tested Components

The most analysed component in the literature is the roller bearing, since it is a critical component for rotating machinery, as explored in Section 3.1. Due to this fact, more information is available pertaining to simulating or replicating faults in this type of component. This led to the choice of replicating faults in a ball bearing in order to collect samples of anomalous behaviour of the system. Figure 4.21 displays the bearing in normal operating conditions, the top picture shows the bearing with the protective caps, which prevent debris from entering the inside structure and also prevent loss of lubricant, while the bottom picture reveals the inside of the bearing.

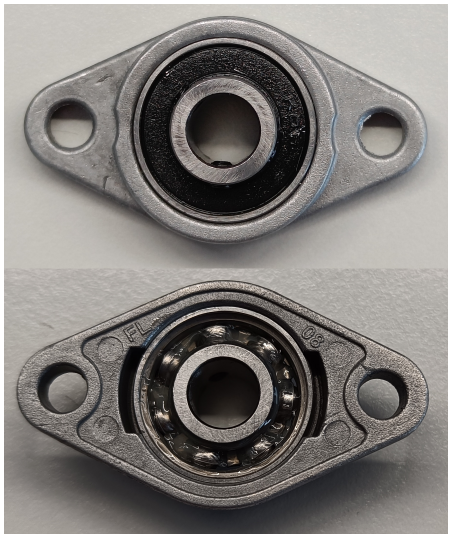


Figure 4.21: Ball Bearing under testing in normal operating conditions

Figure 4.22 features the alterations done to the normal condition ball bearing: removal of most of the lubricant and mechanical indentation. These methods are one of the most common for creating defects on bearings [40].

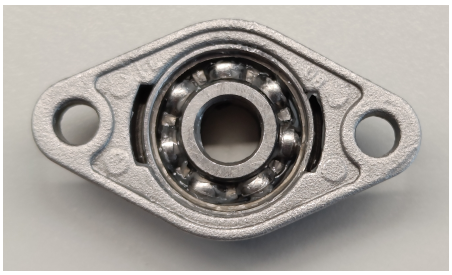


Figure 4.22: Ball Bearing under testing with lubricant partially removed

## 4.4 Software Structure

In this section, the software structure for all parts of this project will be detailed, from the data collection portion, reviewing the algorithms used for the Teensy 4.1, passing through the dataset creation, up to the ML models implementation.

### 4.4.1 Data Collection

To recapitulate from Section 4.1, the chosen sensors were the following:

- **Accelerometers**: *ADXL355* and *MMA8451*
- **Microphones**: *SPH0645LM4H* and *SPU0410LR5H-QB*
- **IR Thermometer**: *MLX90614*

All the digital sensors except for the microphone possess libraries for the Arduino framework, which greatly facilitates the software implementation since it minimises the direct interaction with the SPI and I2C protocols as well as with direct register manipulation. Such low-level coding approaches are inherently error-prone and less human-readable when compared to higher-level code, making the availability of Arduino libraries a significant advantage in ensuring robust software. The digital microphone uses a different protocol, I2S. Thankfully, the Teensy platform provides a library for interacting with I2S devices using Teensy microcontrollers, the *Teensy Audio Library*[17], so it is possible to abstract the final code from directly managing I2S operations.

Before utilising the ESP-12E for sending data wirelessly, it was decided that it was better to record every sample and to save onto an SD card, made possible by an integrated SD card reader available in the Teensy 4.1. This enables saving the raw samples and testing different pre-processing techniques. It also removes the need to implement an application layer, redirecting the effort on analysing the data.

The objective of the microcontroller's program is to read the data from the sensors and to store the measurements in the provided SD card. For convenience reasons, the data from each sensor should be stored in a separate file or folder. For running the higher order operations of the data collection algorithm, a finite state machine was developed, in Figure 4.23. Two push buttons

[17] https://www.pjrc.com/teensy/td_libs_Audio.html

were connected to the microcontroller to interact with the program: one to record a single one-second sample and a different one to start a loop of recording and saving samples. These buttons, combined with the Serial port, enabled debugging each stage. The finite state machine in Figure 4.24 controls the looping of the program.

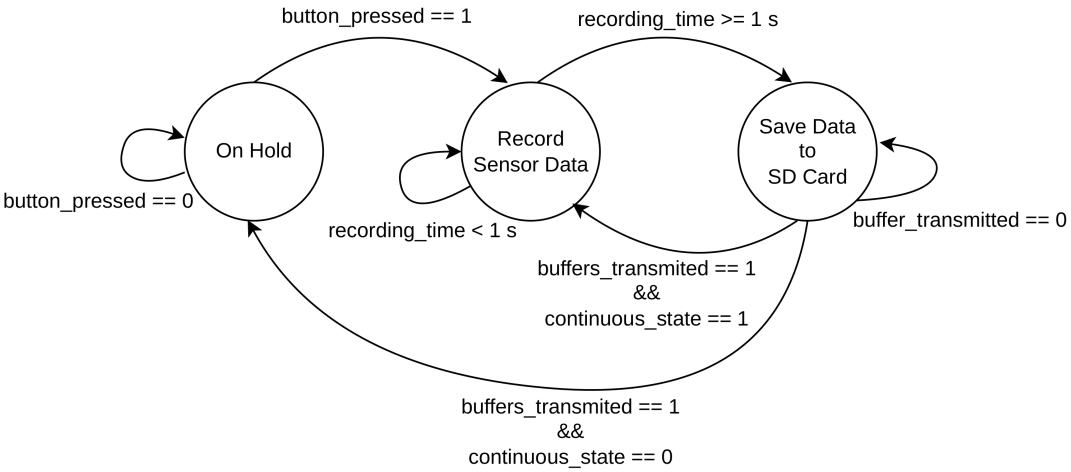



Figure 4.23: Main Finite State Machine - SD card version

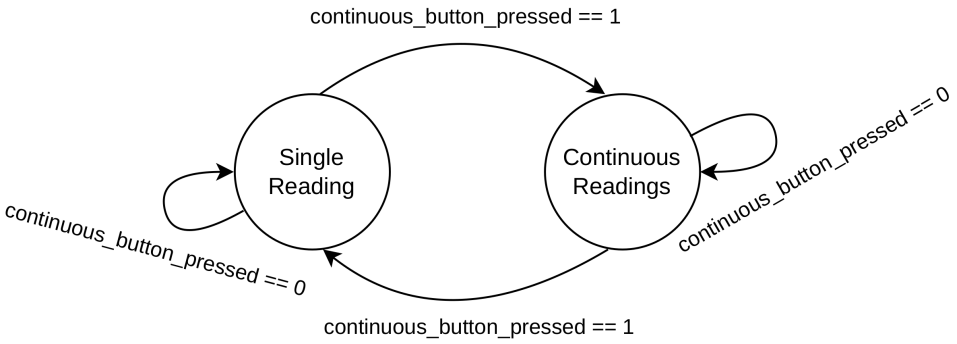



Figure 4.24: Continuous Recording Finite State Machine

The finite state machines control the higher-order operation of the microcontroller, but the state of recording and saving the data requires more detail since it is necessary to maintain a fixed and rigorous sampling frequency for each sensor. Figure 4.25 shows a flowchart of the reading stage. For reading from the digital I2C and SPI sensors, the following libraries were used:

- https://github.com/DFRobot/DFRobot_MLX90614
- https://github.com/adafruit/Adafruit_MMA8451_Library
- https://github.com/plasmapper/adxl355-arduino/

These libraries simplify the interaction with the sensors as the library's user only needs to call one or two functions to get the values. However, the libraries do not guarantee the necessary

sampling rates, so an extra library was utilised to control the hardware timers available in the Teensy 4.1: https://github.com/luni64/TeensyTimerTool.

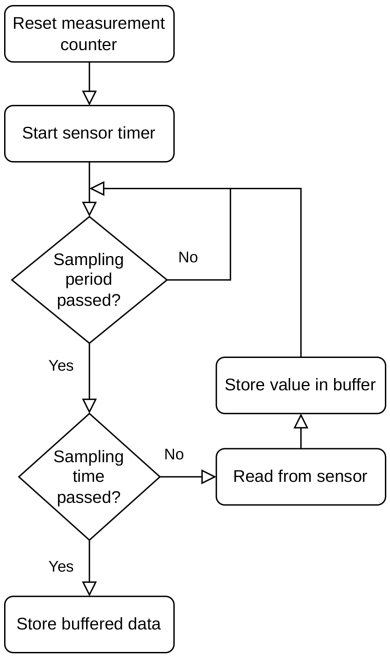



Figure 4.25: Read sensor data Algorithm Flowchart

The final digital sensor, the *SPH0645LM4H* microphone, uses the I2S protocol. It requires a specific hardware controller, which is present in the Teensy 4.1, and an integrated library can be used: *Audio.h* [18]. The library uses the *Teensy Audio System Design Tool* [19] to simplify the setup of different audio devices. The tool uses a block diagram to represent input sources, audio filters, amplifier stages, audio synthesizers, etc., as the Teensy platform is often used for audio projects, and turns the diagram into code, automatically. The blocks used for setting the I2S microphone as an input device and to record the output to a buffer:

- *i2s1* - I2S input
- *filter1* - filter stage
- *amp1* - amplifier stage
- *queue1* - output

The code to record audio to an SD card was based on the example provided by the *Audio.h* library, *Recorder.ino*. It records 512 bytes of audio before writing to an open file in the SD card until 1 second has passed. The microphone outputs a binary file format, *.raw*, which has to be later encoded into an audio format if playback is desired.

[18] https://github.com/PaulStoffregen/Audio
[19] https://www.pjrc.com/teensy/gui/index.html

Concerning the analog microphone, to achieve the needed sampling rate of twice the bandwidth, according to the Nyquist Sampling Theorem, meaning a sampling frequency of 160 kHz, the normal usage of the ADC with the *ADC.h* library [20] is not possible. The *analogRead.ino* example that is provided, in the fastest setting, would take about 13 $\mu$s to take each sample, while a 160 kHz sampling frequency requires a sample every 6.25 $\mu$s, which makes it impossible to use the normal ADC mode in this use case scenario. Thankfully, the library enables accessing the ADC using Direct Memory Access (DMA), which can drastically decrease the time of retrieving the samples. The final implementation of the analog sampling algorithm was based on the *adc_timer _dma.ino* example, which continuously fills a buffer with the samples taken from the signal and when the timer set frequency is met, an interrupt is made, and the values from the buffer are passed to the memory of the microcontroller. Using DMA, each operation to retrieve the samples from the buffer, which contained 1600 samples, takes less than 1 $\mu$s, enabling the required sampling frequency to be achieved.

For saving the data into the SD card, the *SD.h* library in the Arduino Framework is used. It utilises SPI for interfacing with the card and its storage. Figure 4.26 contains a flowchart showing the algorithm employed to store the measurements (contained in buffers). Character strings containing the file names are modified after each sample is taken to reflect the number of the sample so that it is possible to join each sensor's second worth of data together. Apart from the *SPH0645LM4H* microphone, the readings are stored in *.csv* files making it easier to separate the measurements of different axes, in the case of the accelerometers. Writing *.csv* files does not need a different method from writing *.txt* files, as it is a text format type file and the only needed step is to separate each axis with a comma and each instance of the buffer with a paragraph.

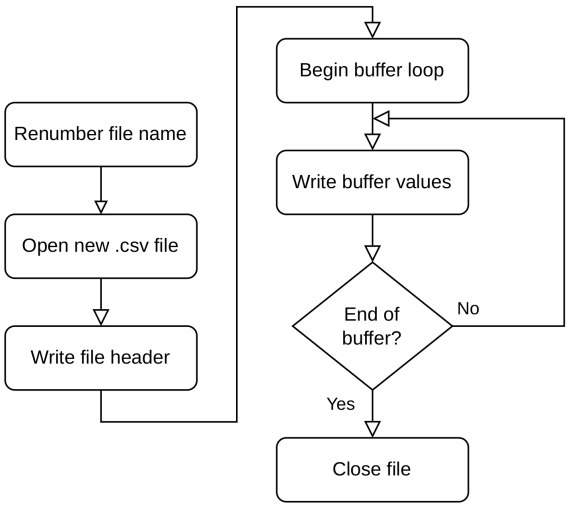



Figure 4.26: Save data in SD card Algorithm Flowchart

[20] https://github.com/pedvide/ADC

### 4.4.2 Dataset Creation

In the last Subsection, the methods required to collect the relevant data for the project's objective were reviewed, but steps remain before using the data to train ML models to detect anomalies. This Subsection addresses these procedures, which include Preprocessing and the merging of the different types of data, vibration, sound and temperature.

#### 4.4.2.1 Preprocessing Steps

As discussed in Section 3.4, different types of data require different processes for extracting the most relevant information from them. In this dissertation, particular emphasis has been placed on vibration and sound data. This is due to these data types' high variability in the context of mechanical maintenance. Temperature rises in much longer time frames and more consistently when compared to vibration or sound. The chosen method is composed of two steps: the scaling of the data and time domain feature calculation. The first step was taken from [50], where acceleration data is passed through the *RobustScaler* function from *sklearn*[21]. This scaler method removes the median and scales the data according to the quantile range, which was chosen to be between quantiles 5 and 95, as per the cited article. For the temperature, the *StandardScaler* [22] was used, since this type of data is not prone to such high variance as vibration and sound. The second step is described in [83], where central tendency-based time domain features are computed before using them as the input for an ML model. The utilised characteristics are defined in the following Equations 4.2 to 4.9, where $X$ represents a sample.

$$RMS = \left( \frac{1}{N} \sum_{i=1}^{N} [X(i)^2] \right)^{\frac{1}{2}} \tag{4.2}$$

$$Variance(\sigma^2) = \frac{1}{N} \sum_{i=1}^{N} (X(i) - \mu)^2 \tag{4.3}$$

$$Skewness = \frac{1}{N} \sum_{i=1}^{N} \left( \frac{X(i) - \mu}{\sigma} \right)^3 \tag{4.4}$$

$$Kurtosis = \frac{1}{N} \sum_{i=1}^{N} \left( \frac{X(i) - \mu}{\sigma} \right)^4 \tag{4.5}$$

[21] https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.RobustScaler.html
[22] https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html

$$CrestFactor = \frac{max(|X|)}{RMS} \tag{4.6}$$

$$ImpulseFactor = \frac{max(|X|)}{\frac{1}{N}\sum_{i=1}^{N}|X(i)|} \tag{4.7}$$

$$ShapeFactor = \frac{RMS}{\frac{1}{N}\sum_{i=1}^{N}|X(i)|} \tag{4.8}$$

$$Range = max(X) - min(X) \tag{4.9}$$

This method was chosen over more traditional steps, e.g. the FFT, since it is simpler to implement and these calculations could be easily transferred to the microcontroller, which could reduce the amount of data to be sent wirelessly. Using the FFT implies selecting a certain number of frequency bins, which can be a very large number when compared to the selected method. For example, in [50], 2048 per 1-second sample is the chosen value, based on the approximate accelerometer bandwidth of 2 kHz. Since, in this project, 3-axis accelerometers were chosen, this number would be multiplied by three, while using the previous Equations, the total number of values to be sent is 24 for each accelerometer. Another possible advantage lies in potentially being less reliant on the bandwidth of the sensors, as sensors with lower bandwidth tend to be of lower cost. Although in [83], these preprocessing calculations in exclusively in vibration data, due to the similar nature of vibration data and sound data, which was previously explored, Equations 4.2 to 4.9 where also applied to the sound data.

#### 4.4.2.2 Dataset Merging

To employ a single ML model with all the data types, merging the datasets is necessary to create the training data. The primary challenge lies in that each sensor captures different sample quantities of data for the same 1-second sample. For example, the ADXL355, according to its datasheet [120], has a bandwidth of 1 kHz, meaning at least two thousand data points (due to the Nyquist Theorem), are recorded, for three different axes, while only one temperature measurement is made by the MLX90614. The method that was found to be the most suitable is the one used in [83], which is previously detailed in Subsection 3.4.1. The strategy lies in using time-domain characteristics of the vibration and sound signals, such as the variance and kurtosis. By utilising these features, it is possible to describe the measured signal through variables that represent 1 second's worth of data. The selection of sample duration aligns with the methodology presented in [83], wherein 10-second samples are initially acquired and subsequently partitioned into 0.25-second segments.

The decision to divide the samples by a factor of 40 is rooted on the fact that the motor used in the testbed rotates the main shaft at 29 Hz, meaning each 0.25-second excerpt contains more than 7 revolutions of the shaft which was considered to be sufficient for computing trustworthy statistical features. In the context of this dissertation, the exact rotation speed of the shaft is unknown, while it is over 7 Hz, so 1 second is a suitable time frame to capture credible measurements within the scenario.

Figure 4.27 shows some features and the structure of the final dataset. The suffix *_x*, *_y* or *_z* represents the axis of the feature. The prefix *355* refers to the *ADXL355* accelerometer, *mma* to the *MMA8451* accelerometer, *digital_mic* to the *SPH0645LM4H* microphone and *object* to the object's measured temperature using the *MLX90614* IR thermometer.

| | 355_crest_factor_x | mma_variance_x | digital_mic_kurtosis | object |
|---|---|---|---|---|
| 0 | -0.279419 | -0.233221 | 0.267974 | -4.043641 |
| 1 | -0.215864 | 0.217168 | 0.095843 | -3.734514 |
| 2 | -0.230050 | 0.105980 | 0.331499 | -3.726786 |
| 3 | -0.332452 | -0.071428 | 0.189802 | -3.842709 |
| 4 | 0.176126 | -0.043893 | 0.150345 | -3.943175 |
| ... | ... | ... | ... | ... |
| 4043 | -0.277202 | -0.161779 | 0.233973 | 0.956483 |
| 4044 | 0.071946 | -0.238543 | -0.010585 | 0.879201 |
| 4045 | -0.319090 | -0.315115 | 0.161555 | 0.740094 |
| 4046 | -0.269076 | -0.248959 | 0.112263 | 0.987395 |
| 4047 | -0.032778 | -0.187514 | 0.034154 | 0.685997 |

Figure 4.27: Dataset Structure

### 4.4.3 ML Models

As previously described, the problem to be solved is Anomaly Detection, where the objective lies in finding data that deviates from the majority. In this context, an anomaly constitutes a mechanical fault that causes the system to behave differently from the *normal* state. This behaviour manifests through different vibration patterns, which leads to an increase in temperature of the damaged component. As the AGV comprises several critical components, it's important to acknowledge that the built testbed can only simulate a subset of potential anomalies. Consequently, it's not feasible to capture data representing every possible anomaly scenario. This limitation suggests that the most suitable type of ML model to employ would be a Semi-Supervised model, which theoretically possesses the capability to classify a new sample that deviates from the characteristics present in the training data, labeling it as an anomaly. However, a significant portion of successful Semi-Supervised Learning algorithms are rooted in Deep Learning, which typically demands substantial datasets (comprising tens of thousands of data points) and can be more complex to implement. Due to time constraints, the project focused exclusively on Supervised and Unsupervised ML models.

#### 4.4.3.1 Supervised Models

In the selection of Supervised ML models, our choices were primarily guided by prior research findings and their availability within the *sklearn* package or their ease of implementation. The final set of models includes:

- k-Nearest Neighbours (k-NN) [23]
- Support Vector Machine [24]
- Random Forest [25]
- AdaBoost [26]
- XGBoost [27]

It is worth noting that all models, except XGBoost, are readily available through *sklearn*. Some models are not covered in Chapter 3 because no relevant articles were found focusing on them.

#### 4.4.3.2 Unsupervised Models

Following the selection criteria for the Supervised ML models, the Unsupervised ML model set:

- One-Class SVM [28]
- Isolation Forest [29]
- Local Outlier Factor (LOF) [30]

It is important to highlight that the selection of Unsupervised models is intentionally narrower compared to the Supervised models. This decision stems from the belief that, in this use case, it is crucial to have the capability to classify data as normal behaviour. While Unsupervised models were still tested to account for unpredictable outcomes.

[23] https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html
[24] https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html
[25] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
[26] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html
[27] https://xgboost.readthedocs.io/en/stable/python/index.html
[28] https://scikit-learn.org/stable/modules/generated/sklearn.svm.OneClassSVM.html
[29] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.IsolationForest.html
[30] https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.LocalOutlierFactor.html

# Chapter 5

# Experimental Results

In continuation of the processes outlined in the preceding chapter, the subsequent chapter outlines the techniques employed to assess the chosen models, while considering various scenarios, including sampling restrictions.

## 5.1 Metrics

For evaluating the effectiveness of the ML models in detecting anomalies in the datasets, it is necessary to choose adequate metrics that discern the most pertinent aspects of the model's performance for the associated scenario. Recapitulating, the ML problem is a binary classification question: if the sample is of anomalous behaviour, then the *anomaly* feature is a *1*; if it is normal behaviour, the *anomaly* feature is a *0*.

For classification problems, one of the most popular methods for evaluation of the model is by using a confusion matrix. This $N \cdot N$ matrix, where $N$ represents the number of target classes, displays the actual target values with those predicted by the model. Figure 5.1 [1] shows an example of a confusion matrix: the top left corner is the case where the *true* label for the data point was *0* and the model correctly predicted that label; the top right corner represents the mislabelling of the *0* class; the bottom left corner is the mislabelling of the *1* class; the bottom corner is the correct labelling of the *1* class. From these four values, it is possible to extract the number of *true positives*, *false positives*, *true negatives* and *false negatives*.

Other metrics have emerged from these four base numbers whose objective is to highlight certain aspects of the model. In the context of this dissertation, the focus lies on minimising *false negatives*, where an anomalous sample is incorrectly classified as normal, as this misclassification could potentially lead to an unscheduled maintenance stop for the AGV and a halt of production, which is less favourable than an unnecessary scheduled maintenance stop. Keeping this goal in mind, the following metrics were employed:

- Recall

[1] Taken from https://scikit-learn.org/stable/modules/generated/sklearn.metrics.ConfusionMatrixDisplay.html

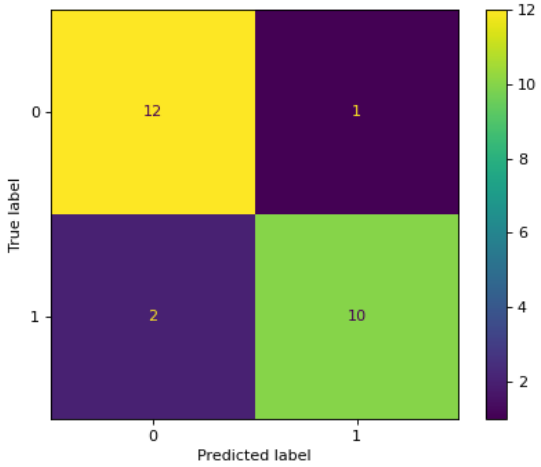



Figure 5.1: Confusion Matrix Example

- Precision
- F1-Score
- Accuracy

To underscore the importance of reducing false negatives, *recall* (as defined in Equation 5.1 [2]) was picked. A *recall* value closer to 1 signifies a higher rate of true positives relative to false negatives, highlighting the model's ability to correctly identify anomalies. *Precision*, defined in Equation 5.2, highlights the fraction of relevant instances among the retrieved instances, meaning a value closer to 1 is a better score than a value closer to 0. *F1 Score* is defined in Equation 5.3 and takes into account *Recall* and *Precision* and can be understood as the harmonic mean of the two. The *F1 Score* reaches its best value at 1. The final metric is *accuracy* and encompasses predicted labels that exactly match the true labels, calculated using Equation 5.4.

$$Recall = \frac{TP}{TP + FN} \tag{5.1}$$

$$Precision = \frac{TP}{TP + FP} \tag{5.2}$$

$$F1\,Score = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{5.3}$$

[2] TP - True Positives; TN - True Negatives; FP - False Positives; FN - False Negatives

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{5.4}$$

## 5.2 Test Structure

In this section, the structure for the evaluation of the models will be overviewed. The goal of producing a custom dataset was to validate the approach taken, concerning the chosen sensors, pre-processing steps and ML models, while addressing a setting that is not frequently researched in the field. The decision to build three different testbeds allows for comparing the same models trained with different data and to control what anomalies are recorded, the environmental noise and the amount of samples to obtain.

### 5.2.1 Dataset Unbalance

As previously stated, one of the challenges of Anomaly Detection is the potential lack of samples of anomalous data due to anomalies representing rare events which can be difficult to capture. A benefit of having a custom dataset is the ability to control the amount of anomalies present in the training dataset in relation to the normal samples. Testing the models with different percentages of anomalies in the dataset allows for simulating a scenario where anomalous behaviour is hard to capture and comparing the model's performance when being provided with an unbalanced dataset. It is important to note that some of the selected models, namely *Isolation Forest* and *Local Outlier Factor* expect that the dataset is unbalanced and one of the input parameters is the fraction of anomalous data.

Different amounts of samples were recorded from each testbed:

- **Testbed 1**: 3018 samples of *normal* behaviour and 3058 samples of *anomalous* behaviour;
- **Testbed 2**: 4049 samples of *normal* behaviour and 4323 samples of *anomalous* behaviour;
- **Testbed 3**: 4161 samples of *normal* behaviour and 3966 samples of *anomalous* behaviour;

The disparity between the testbeds occurred due to initial software issues. These manifested as the finite state machine becoming stuck in the recording function of the *SPH0645LM4H* I2S microphone after approximately 100 samples had already been saved. The software was adjusted to mitigate the issues, reducing the number of stops to restart the program, after extracting the samples from the SD card.

For creating unbalanced datasets, the following percentages of anomalies were chosen: 50%, 43%, 33%, 20% and 9%. These values were selected *a posteriori*, following manually entered amounts and an analysis of the impact on model performance using these thresholds. These percentages were subsequently employed to automate the model testing procedures, which will be explored ahead.

### 5.2.2 Sensor Comparison

One of the objectives of this dissertation is for the sensor module to be cost-effective, which is the justification for integrating two different accelerometers in the testing procedures: the *ADXL355* presents more adequate characteristics for the subsequent analysis, according to the research done for the project, but the *MMA8451* is, at the time of writing, more than six times cheaper. The custom datasets can combine the readings from all the integrated sensors, combine certain sensors or use the readings from a single sensor. This enables the comparison of multiple combinations which adapt to the problem at hand: comparing the two accelerometers, comparing the most expensive sensors with a junction of the cheapest and comparing the impact of each sensor on the model's performance, which was also taken into account in the automation of the testing procedures.

### 5.2.3 ML Automation Script

*Python* was the language of choice for creating and testing the ML models. One advantage of using it is the ability to create a list or array element containing any type of variable, including *sklearn* objects and any other object from different packages. This enables iterating through models and retraining them using different percentages of anomalies in the datasets while saving the relevant metrics and images of each confusion matrix for each scenario.

Figure 5.2 displays the process, through a flowchart, used in the script for automating the training and testing of each models while modifying the dataset provided with different numbers of anomalies, as described in SubSection 5.2.1, and computing and saving the metrics discussed in Section 5.1.

The script possesses two different parts, one for the Unsupervised models and another for the Supervised ones, since these two kinds of models require slightly different procedures, as the Unsupervised models do not require a training procedure. For using the script with different sensor data, it is only required to alter the *.csv* file containing the data. All Supervised models were trained and tested with a *Train Test Ratio* of 80/20, meaning 80% of the data was used for training the models and 20% for testing. It is also important to note that it is assumed that the preprocessing steps have already been performed and that the provided file contains the preprocessed data.

## 5.3 Ultrasonic Microphone Measurements

One of the primary challenges encountered during the development of the sensor module revolved around the SPU0410LR5H-QB ultrasonic microphone, particularly in terms of amplifying its signal and converting it into the digital domain. Unfortunately, the measurements proved to be unusable, as evidenced by the complete saturation of the ADC's input, as illustrated in Figure 5.3. During testing, the input was not saturated, and the values were centred around 2048 (in line with the offset of 1.65 V given to the output of the amplifier).

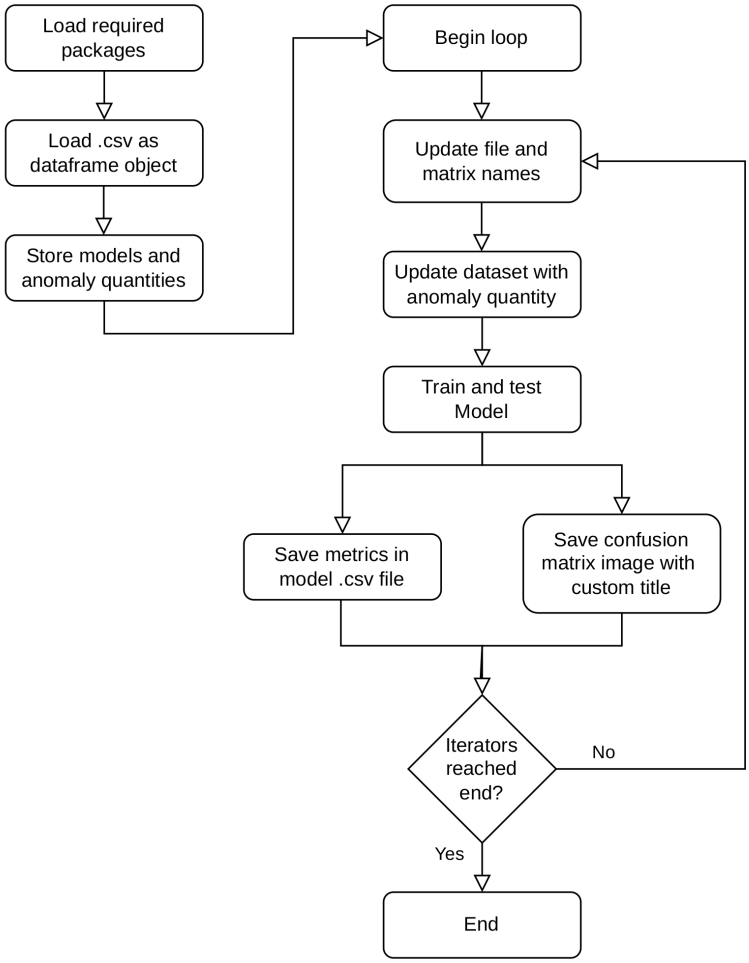



Figure 5.2: ML Automation Script Flowchart

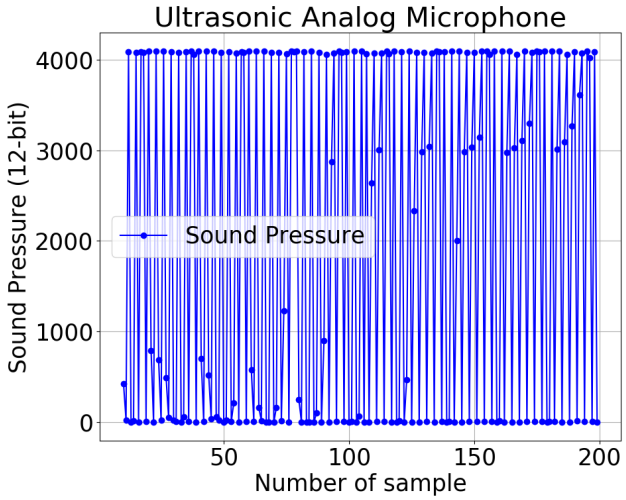



Figure 5.3: Ultrasonic Microphone Readings - 0.5 second sample

Figure 5.4 shows a more zoomed-in perspective of the sampled signal, where it is possible to observe that not all values are either 0 or close to 4000 (the limits of the input range).

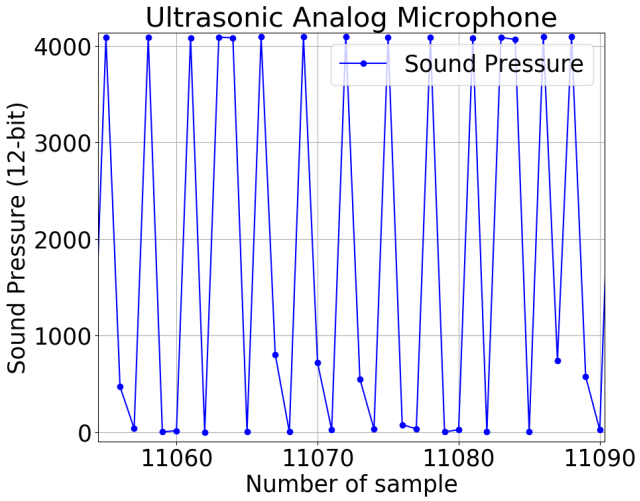



Figure 5.4: Zoomed In Ultrasonic Microphone Readings

Comparing the difference between the microphone's testing setup and the sensor module's setup, the difference lies in the number of sensors connected to the power lines of the microcontroller, so the signal might be contaminated by noise in the power supply. The amplifier's design was based on the one in [131], but the one employed in this dissertation does not have an electrical noise isolation stage nor an external voltage reference chip.

The observation of this error was almost immediate, and the ultrasonic microphone circuit was removed from the module and its measurements were not included in the training of the ML models.

## 5.4 Temperature Analysis

The data was sampled sequentially, starting from Testbed 1. Then, the data was analysed and the *Python* scripts for preprocessing and for concatenating were made and tested on the data. This sequence, as opposed to recording all the Testbeds and then make the scripts, was chosen in order to optimise the process of acquiring the samples and to check what parts of the whole process would take up the most time, building the final workflow. The easiest variable to process both computationally and visually is temperature. Figure 5.5 displays the temperature readings from Testbed 2 of the *normal* bearing, from sample 1450 up to sample 4030. In orange, the temperature read from the IR thermometer and, in blue, the ambient temperature. A constant rise in the temperature up to a steady state is noticeable and explained by the fact that recordings began while the components were at ambient temperature.

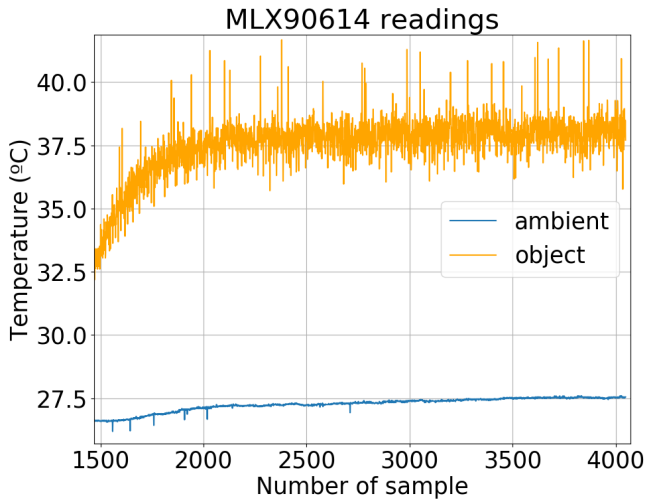



Figure 5.5: Temperature Readings from Testbed 2 - Sample 1450 to 4030

A different aspect is also noteworthy and emerges by comparing the difference between two consecutive readings of the bearing's temperature (Figure 5.6 shows a zoomed-in perspective of the readings): the temperature fluctuates a relatively high amount within seconds apart, while the ambient temperature's fluctuation is much smaller. It can be deduced that the IR thermometer was not measuring the same point on the bearing surface, probably due to being attached to the wooden platform, which vibrated due to the components being tested. This error in the method of measurement could possibly have been solved by using a different board for the sensor with a smaller field of view since the one used from *seeed studio* has a field of view of 35º or by using a more secure attachment method.

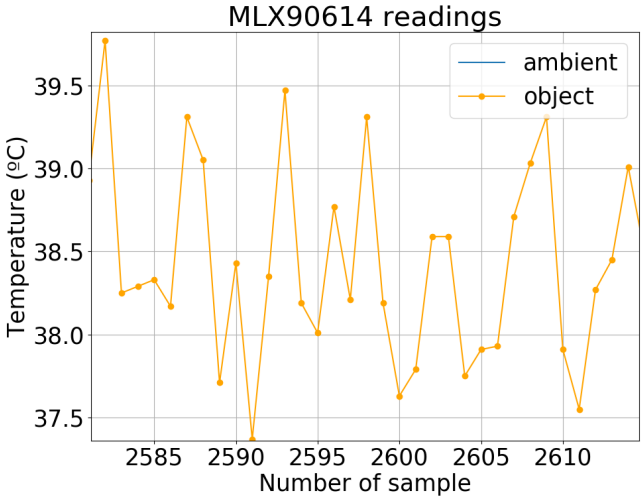



Figure 5.6: Zoomed In Temperature Readings from Testbed 2

An analysis perspective that contrasts the initial reasoning point for temperature as a measurement unit to use to detect anomalies is that it is not useful for the objective. Temperature changes

much more slowly than vibration or sound, and it would require for the AGV to continuously run for its critical components' temperature to increase significantly before a fault is detected, while vibration and sound signals are always affected by damaged components. This is perhaps the reason for the difference in the number of articles which use vibration or sound for PdM, while temperature is not as often utilised.

The previous speculation is demonstrated in the results of the models when removing temperature from the dataset. Table 5.1 compares metrics of 3 models trained (k-NN, SVM and Random Forest) with all sensors and with all sensor data apart from temperature. In these examples, the exclusion of temperature has a negligible impact on the results and the differences of less than 1% could be attributed to the limited dataset size or the inherent randomness inherent in ML models.

Table 5.1: Comparison of models using all sensors and without temperature data - Testbed 1

| Model | Sensors | Anomaly (%) | Recall | Precision | F1-Score | Accuracy |
|---|---|---|---|---|---|---|
| k-NN | All | 50 | 0.99 | 0.99 | 0.99 | 0.99 |
| | No temperature | | 0.99 | 0.99 | 0.99 | 0.99 |
| SVM | All | | 0.98 | 0.99 | 0.98 | 0.98 |
| | No temperature | | 0.98 | 0.98 | 0.98 | 0.98 |
| RF | All | 33 | 1 | 1 | 0.99 | 0.99 |
| | No temperature | | 1 | 1 | 1 | 1 |

Since adjustments were made between recordings and due to this lack of impact of temperature in the results, temperature was not recorded for Testbed 3.

## 5.5 Unsupervised Models Results

In this Section, the metrics of the Unsupervised Learning Models will be analysed. To recapitulate, three models were selected: One-Class SVM, Isolation Forest and Local Outlier Factor. In Table 5.2, the best results for all combinations are displayed, while excluding F1-Score. Many of the models exhibited performance akin to that of the Testbed 1 version of the One-Class SVM, where their accuracy was only marginally superior to that of random guessing. However, some models (e.g. Testbed 2 Isolation Forest with 9% anomalies) possess relatively high accuracy and recall values, but the precision values are very low. Figure 5.7 shows the confusion matrix of the previous example and demonstrates the statement, as the model classifies almost every sample as *normal*, which explains high accuracy and recall since only 9% of all samples to classify are *anomalous*. This displays the benefit of using multiple metrics to evaluate a model's performance.

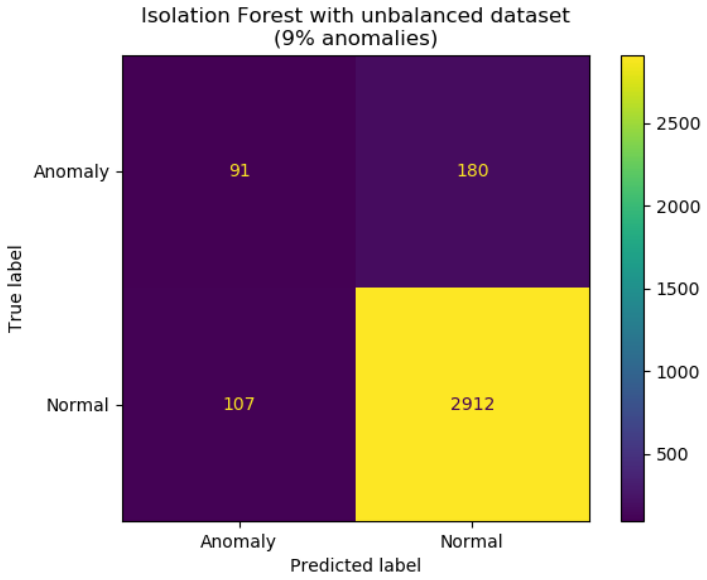



Figure 5.7: Isolation Forest Confusion Matrix - 9% anomalies

The poor performance of the Unsupervised models could potentially be attributed to the choice of pre-processing steps employed on the data. To enhance their performance, alternative approaches could be explored, such as frequency-based or time-frequency-based methods, as discussed in Chapters 2 and 3.

## 5.6 Supervised Models Results

This section will go further in the exploration of the results of the Supervised models than in Section 5.5 since the metrics are objectively superior. Table 5.3 details the best models in the most challenging conditions, using all features except temperature: 9% of anomalies in the dataset. The results are nearly ideal, while it is necessary to consider the small sample size, within the case of Testbed 2, 109 anomalous samples, as shown in Figure 5.8.

Table 5.2: Unsupervised Models Metrics

| Model | Testbed | Sensors | Anomaly (%) | Recall | Precision | Accuracy |
|---|---|---|---|---|---|---|
| OC SVM | 1 | All | 50 | 0.59 | 0.59 | 0.59 |
| | 2 | All | | 0.46 | 0.45 | 0.46 |
| | 3 | No temperature | | 0.46 | 0.45 | 0.45 |
| I-Forest | 1 | All | 20 | 0.87 | 0.34 | 0.81 |
| | 2 | All | 9 | 0.91 | 0.17 | 0.85 |
| | 3 | No temperature | | 0.90 | 0.060 | 0.83 |
| LOF | 1 | All | 9 | 0.91 | 0.073 | 0.87 |
| | 2 | All | | 0.91 | 0.14 | 0.85 |
| | 3 | No temperature | | 0.93 | 0.34 | 0.89 |

Table 5.3: Best Results from Supervised Models

| Model | Testbed | Sensors | Anomaly (%) | Recall | Precision | F1-Score | Accuracy |
|---|---|---|---|---|---|---|---|
| XGBoost | 1 | No temperature | 9 | 1 | 1 | 0.99 | 0.99 |
| | 2 | No temperature | | 1 | 1 | 0.99 | 0.99 |
| | 3 | No temperature | | 1 | 1 | 1 | 1 |

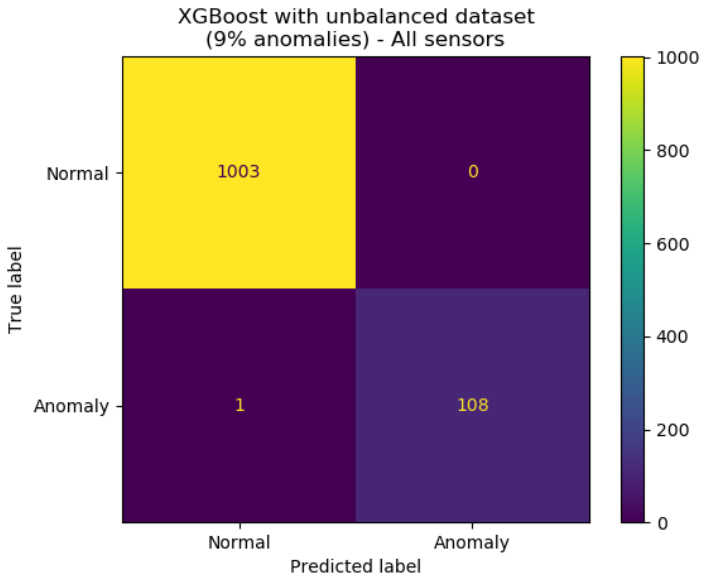



Figure 5.8: Testbed 2 - XGBoost Confusion Matrix - 9% anomalies

It is necessary to take into account that Table 5.3 shows the model with the best results, namely XGBoost. Table 5.4 shows the performance of the other models under the same conditions. Using four different metrics proves to be a valuable approach, given that three of these metrics are consistent across all models, with the exception of the F1-Score, which demonstrates markedly inferior results in the SVM and Random Forest models. This multi-metric evaluation structure allows for a more comprehensive assessment of each model and its respective performance.

### 5.6.1 Sensor Comparison

As emphasised throughout this dissertation, a key goal is to ensure the cost-effectiveness of the module. Therefore, it becomes essential to evaluate the performance of various sensors while considering their cost. The ML Automation Script allows for simple testing of multiple alternatives: using only the accelerometers or only the microphone or by combining different sensors. Table 5.5 presents a selection of models along with the corresponding training data. The models chosen were the ones were the most variation occurred between evaluation metrics.

Table 5.5 shows that the better-performing models only present slight variations when changing the sensors used in the training data. However, the ones with previous inferior performance

Table 5.4: More results for the same conditions

| Model | Testbed | Sensors | Anomaly (%) | Recall | Precision | F1-Score | Accuracy |
|---|---|---|---|---|---|---|---|
| k-NN | 1 | No temperature | 9 | 1 | 1 | 0.97 | 0.99 |
| | 2 | | | 0.98 | 0.99 | 0.98 | 0.99 |
| | 3 | | | 1 | 1 | 0.81 | 0.97 |
| SVM | 1 | | | 1 | 1 | 0.62 | 0.95 |
| | 2 | | | 1 | 1 | 0.083 | 0.90 |
| | 3 | | | 0.95 | 0.99 | 0.39 | 0.93 |
| RF | 1 | | | 1 | 1 | 0.97 | 0.99 |
| | 2 | | | 1 | 1 | 0.55 | 0.94 |
| | 3 | | | 1 | 1 | 0.74 | 0.96 |
| AdaBoost | 1 | | | 1 | 1 | 1 | 1 |
| | 2 | | | 0.99 | 0.95 | 0.95 | 0.99 |
| | 3 | | | 1 | 1 | 1 | 1 |

display much higher variation, especially by only using either the *MMA8451* accelerometer or the *SPH0645LM4H* microphone, e.g. SVM's accuracy goes down to 0.57 and 0.78, respectively. One possible way to improve the model performance using these sensors is to combine the data from both. Table 5.6 displays metrics for the worst performing models in Table 5.5 being retrained with data from both sensors, while maintaining the conditions of the testbed and anomaly percentage.

Employing only the *ADXL355* accelerometer for the training data produces marginally worse performance for almost every scenario than using all sensor data. The combination of the cheaper sensors (*MMA8451* and *SPH0645LM4H*), which total in price of 13,85 € [128] [122] [3], also produces near-ideal results.

### 5.6.2 Unknown Anomalies

While the previous results have shown that some Supervised models have produced near-perfect predictions in detecting anomalous behaviour based on 1-second samples. However, the recorded anomalies do not represent every possible anomaly that might occur in a real scenario. Since three different testbeds were used, it is possible to join datasets to *simulate* new anomalies by using data from a different testbed than the one used for training to test the models and evaluate their performance when facing anomalies that were not represented in the dataset. Table 5.7 presents the best model to have met these conditions, it was trained using data from Testbed 3 and tested using anomalies from Testbed 2. Figure 5.9 shows the confusion matrix of the same model. Automating tests would provide more insight into this possibility of testing unseen anomalies, but the built script is not capable of taking two different datasets.

The results obtained from this small experiment are promising but further exploration is required to figure out if this method of detecting faults is achievable in a real scenario, particularly when confronted with unforeseen anomalies.

[3] At the time of writing

Table 5.5: Sensor Impact on model performance comparison

| Model | Testbed | Sensors | Anomaly (%) | Recall | Precision | F1-Score | Accuracy |
|---|---|---|---|---|---|---|---|
| XGBoost | 1 | No temperature | 20 | 0.97 | 0.99 | 0.98 | 0.99 |
| | | ADXL355 | | 0.99 | 0.99 | 0.99 | 0.99 |
| | | MMA8451 | | 0.96 | 0.99 | 0.97 | 0.99 |
| | | SPH0645LM4H | | 1 | 1 | 0.97 | 0.99 |
| | 2 | No temperature | | 1 | 1 | 0.99 | 0.99 |
| | | ADXL355 | | 1 | 1 | 0.99 | 0.99 |
| | | MMA8451 | | 0.93 | 0.98 | 0.93 | 0.97 |
| | | SPH0645LM4H | | 1 | 1 | 0.97 | 0.99 |
| K-NN | 2 | No temperature | 33 | 0.98 | 0.99 | 0.98 | 0.99 |
| | | ADXL355 | | 1 | 1 | 0.99 | 0.99 |
| | | MMA8451 | | 0.87 | 0.76 | 0.82 | 0.87 |
| | | SPH0645LM4H | | 0.89 | 0.95 | 0.77 | 0.86 |
| SVM | 2 | No temperature | 50 | 0.94 | 0.95 | 0.91 | 0.92 |
| | | ADXL355 | | 0.9 | 0.91 | 0.79 | 0.81 |
| | | MMA8451 | | 0.74 | 0.75 | 0.79 | 0.78 |
| | | SPH0645LM4H | | 0.65 | 0.8 | 0.45 | 0.57 |
| RF | 3 | No temperature | 33 | 1 | 1 | 0.98 | 0.99 |
| | | ADXL355 | | 1 | 1 | 0.99 | 0.99 |
| | | MMA8451 | | 0.99 | 0.98 | 0.86 | 0.92 |
| | | SPH0645LM4H | | 0.97 | 0.99 | 0.88 | 0.92 |
| AdaBoost | 1 | No temperature | 20 | 1 | 1 | 0.99 | 0.99 |
| | | ADXL355 | | 1 | 1 | 1 | 1 |
| | | MMA8451 | | 0.9 | 0.98 | 0.9 | 0.97 |
| | | SPH0645LM4H | | 0.93 | 0.98 | 0.91 | 0.97 |

## 5.7 Summary

This Chapter delved into the methods and metrics employed to comprehensively evaluate the previously chosen ML models and the operating conditions. One of the main issues in Anomaly Detection, the lack of anomalous data, was addressed by testing the models with different levels of dataset unbalance. More in line with the context of this dissertation, the models were also trained using different sensors to compare performance.

The Unsupervised models performed very poorly, possibly due to the preprocessing steps, while the Supervised models obtained near-ideal results, even with datasets containing 91% of normal data. Having data from three different testbeds allowed for experimenting with classifying previously unseen anomalies. However, more research must be conducted before it is determined the applicability of these tests in a real use case scenario.

Table 5.6: Model metrics combining *MMA8451* and *SPH0645LM4H*

| Model | Sensors | Anomaly (%) | Recall | Precision | F1-Score | Accuracy |
|---|---|---|---|---|---|---|
| SVM | MMA8451 and SPH0645LM4H | 50 | 0.74 | 0.72 | 0.79 | 0.79 |
| RF | | 33 | 0.98 | 0.99 | 0.93 | 0.96 |
| K-NN | | 33 | 0.87 | 0.93 | 0.87 | 0.91 |

Table 5.7: Best results testing with anomalies not present in the dataset

| Model | Sensors | Anomaly (%) | Recall | Precision | F1-Score | Accuracy |
|---|---|---|---|---|---|---|
| RF | No temperature | 33 | 0.59 | 1 | 0.74 | 0.86 |

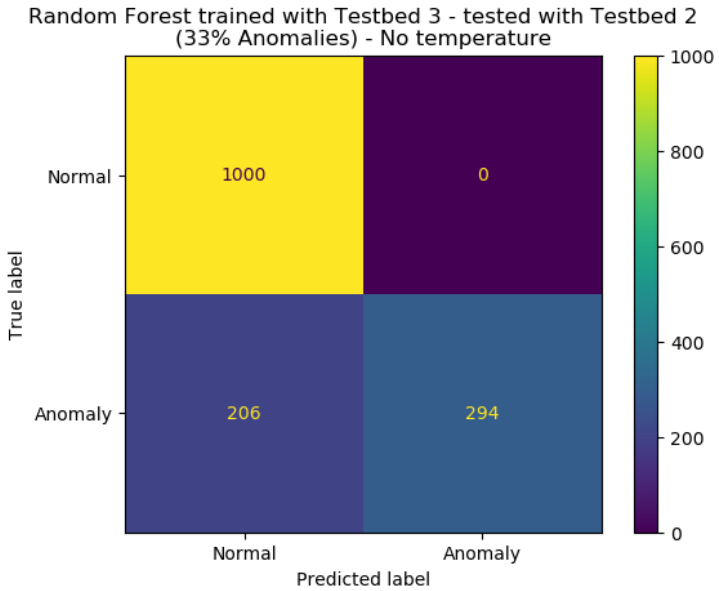



Figure 5.9: Confusion Matrix of model in Table 5.7

# Chapter 6

# Conclusion and Future Work

## 6.1 Conclusion

The objective of this dissertation was to create a cost-effective sensor module for detecting anomalies in an AGV and its surrounding machines. In order to validate the created system, a custom platform was necessary to replicate the AGV, as the existing platforms and public datasets are typically meant for studying industrial machinery, whose physical characteristics are not identical to the AGV.

Three different custom testbeds were built, which allowed for diversifying the collected data and further validating the system, including the hardware and software. The custom platforms can also be used to simulate other types of machines, apart from AGVs, such as machines with smaller traction systems.

A study of the most important features of the sensors employed in public datasets and in the literature enabled a mindful choice of which sensors were the most adequate. This knowledge can be utilised to streamline the sensor selection process, aligning them with a researcher's specific requirements.

The selected accelerometers and the digital microphones proved to be fit for the task at hand, meeting physical space restrictions and providing reliable measurements, which enabled the accuracy of the Supervised models. The IR thermometer and ultrasonic microphone require more preparation and study to determine their possible contribution to the objective.

Regarding the preprocessing of the data and the results, while the preprocessing method employed yielded nearly ideal predictions for the Supervised ML models, the performance of the Unsupervised ML models was notably poor. Therefore, it is advisable to explore alternative techniques, starting from the ones described in Chapter 3. An experiment was made to simulate anomalies not present in the training dataset in order to further test the Supervised models and to evaluate their performance in a possible deployment scenario. Although, in this scenario, the models were less accurate than in the *conventional* scenario, the results proved promising and further research can validate the strategy. It is important to note that the number of collected samples is not sufficient for experimenting with Deep Learning techniques, which are thoroughly reviewed and used

in the literature.

Consideration for an industrial environment was taken throughout the document and thought process, including the environmental noise inherent to a factory, the constricting space limitations of the inside of an AGV and the inability to record every possible fault that might occur in the system.

The ultimate goal of this dissertation was not met, which involved the detection of anomalies in an AGV and in its surrounding machines. However, a strategy for constructing a sensor module capable of achieving the aforementioned objective was delineated and validated, all while considering the constraints inherent to the environmental and spatial conditions.

## 6.2 Future Work

The following section is meant to discuss ideas for improving the work developed in this dissertation. As noted throughout the chapters, PdM is a complex field that encompasses a wide array of research domains, offering ample opportunities for improvement in various aspects of the sensor system, including both hardware and software.

While building the testing platforms, insufficient planning was made for attaching the sensors to the mechanical system, even though it had been recognised that this aspect had a discernible impact on the measurements. For further testing using the AGV itself, more preparation must be done to ensure that the sensors are keeping fidelity in their readings and to try to isolate them from noise as much as possible. Therefore, an enhancement would involve transitioning to either a PCB or soldering the relevant connections. Additionally, using long cables for the sensors would provide greater flexibility for positioning them in optimal locations inside the AGV.

Furthermore, if it is decided to build a more accurate testbed for experimenting with other preprocessing techniques or to fine-tune the sensor module, it is advisable to select components that align more closely with the AGV's specifications, as the one selected for the custom testbeds were only approximately matched.

Using multiple microphones to create a microphone array could potentially enhance the performance of the Anomaly Detection algorithms since it makes it possible to pinpoint the faulty sound signal. This concept has been applied in the field of PdM, namely in [62]. Considering the limited space inside the AGV, this idea requires more exploration in order to name it viable in the target use case scenario.

In this dissertation, calibration of the sensors was not taken into account and relied on the manufacturer's specifications for labelling the measurements as accurate. For future iterations, considering calibration tests would be advisable due to the noise levels present in industrial environments.

As previously discussed, the ideal model type for the problem at hand is Semi-Supervised Learning due to a lack of data of all possible faults of the system. In order to obtain the needed amount of data, changing the recording process would be beneficial since using the one employed in this dissertation takes approximately two hours per 4000 samples captured. A possible approach

is to adapt the algorithm used in [83], where vibration is sampled continuously for 10 seconds and then the data is divided into segments, which would accelerate the recording process. One possible limitation of using this method is the available memory of the microcontroller, which limits the amount of data that can be stored at once before transferring either wirelessly or to external storage.

TinyML [1] is a field of study that brings ML to embedded devices. Implementing a TinyML solution in the prototype would enable the usage of more complex ML models for detecting anomalous behaviour without needing to send data wirelessly, apart from a warning signal that an anomaly has been detected. Furthermore, it would reduce dependence on cloud infrastructure for deploying complex models.

[1] https://www.tinyml.org/

# References


[1] Gil-Yong Lee, Mincheol Kim, Ying-Jun Quan, Min-Sik Kim, Thomas Joon, Young Kim, Hae-Sung Yoon, Sangkee Min, Dong-Hyeon Kim, Jeong-Wook Mun, Jin Woo Oh, In Gyu Choi, Chung-Soo Kim, Won-Shik Chu, Jinkyu Yang, Binayak Bhandari, Choon-Man Lee, Jeong-Beom Ihn, Sung-Hoon Ahn, and William E Boeing. Machine health management in smart factory: A review †. *Journal of Mechanical Science and Technology*, 32(3):987–1009, 2018. ISBN: 1220601802011. doi:10.1007/s12206-018-0201-1.

[2] Patrick Strauß, Markus Schmitz, René Wöstmann, and Jochen Deuse. Enabling of Predictive Maintenance in the Brownfield through Low-Cost Sensors, an IIoT-Architecture and Machine Learning. In *Proceedings - 2018 IEEE International Conference on Big Data, Big Data 2018*, pages 1474–1483. Institute of Electrical and Electronics Engineers Inc., January 2019. doi:10.1109/BigData.2018.8622076.

[3] Tiago Zonta, Cristiano André da Costa, Rodrigo da Rosa Righi, Miromar José de Lima, Eduardo Silveira da Trindade, and Guann Pyng Li. Predictive maintenance in the industry 4.0: A systematic literature review. *Computers and Industrial Engineering*, 150, 12 2020. doi:10.1016/j.cie.2020.106889.

[4] Douglas Thomas and Brian Weiss. Maintenance costs and advanced maintenance techniques in manufacturing machinery: Survey and analysis. *International journal of prognostics and health management*, 12(1), 2021.

[5] Yuki Tagawa, Rytis Maskeliūnas, and Robertas Damaševičius. Acoustic anomaly detection of mechanical failures in noisy real-life factory environments. *Electronics (Switzerland)*, 10(19), October 2021. doi:10.3390/electronics10192329.

[6] Enrico Zio. Prognostics and Health Management (PHM): Where are we and where do we (need to) go in theory and practice. *Reliability Engineering & System Safety*, 218:108119, February 2022. doi:10.1016/j.ress.2021.108119.

[7] ISO Central Secretariat. Condition monitoring and diagnostics of machines — General guidelines. Standard, International Organization for Standardization, Geneva, CH, 2018.

[8] Xiao-Sheng Si, Wenbin Wang, Chang-Hua Hu, and Dong-Hua Zhou. Remaining useful life estimation–a review on the statistical data driven approaches. *European journal of operational research*, 213(1):1–14, 2011.

[9] Dubravko Miljković. Fault detection methods: A literature survey. In *2011 Proceedings of the 34th international convention MIPRO*, pages 750–755. IEEE, 2011.

[10] Rolf Isermann. *Fault-Diagnosis Systems*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006. doi:10.1007/3-540-30368-5.

[11] Rolf Isermann. Model-based fault-detection and diagnosis – status and applications. *Annual Reviews in Control*, 29(1):71–85, January 2005. doi:10.1016/j.arcontrol.2004.12.002.

[12] Dhiraj Neupane and Jongwon Seok. Bearing fault detection and diagnosis using case western reserve university dataset with deep learning approaches: A review. *IEEE Access*, 8:93155–93178, 2020.

[13] Purushottam Gangsar and Rajiv Tiwari. Signal based condition monitoring techniques for fault detection and diagnosis of induction motors: A state-of-the-art review. *Mechanical systems and signal processing*, 144:106908, 2020.

[14] Spilios D Fassois and John S Sakellariou. Time-series methods for fault detection and identification in vibrating structures. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 365(1851):411–448, 2007.

[15] Felix C Freiling, Rachid Guerraoui, and Petr Kuznetsov. The failure detector abstraction. *ACM Computing Surveys (CSUR)*, 43(2):1–40, 2011.

[16] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):1–58, 2009.

[17] Tahereh Pourhabibi, Kok-Leong Ong, Booi H Kam, and Yee Ling Boo. Fraud detection: A systematic literature review of graph-based anomaly detection approaches. *Decision Support Systems*, 133:113303, 2020.

[18] Mohiuddin Ahmed, Abdun Naser Mahmood, and Jiankun Hu. A survey of network anomaly detection techniques. *Journal of Network and Computer Applications*, 60:19–31, 2016.

[19] Stefania Matteoli, Marco Diani, and Giovanni Corsini. A tutorial overview of anomaly detection in hyperspectral images. *IEEE Aerospace and Electronic Systems Magazine*, 25(7):5–28, 2010.

[20] Ashik Mostafa Alvi, Siuly Siuly, and Hua Wang. Neurological abnormality detection from electroencephalography data: a review. *Artificial Intelligence Review*, 55(3):2275–2312, 2022.

[21] Feifei Liu, Chengyu Liu, Lina Zhao, Xiangyu Zhang, Xiaoling Wu, Xiaoyan Xu, Yulin Liu, Caiyun Ma, Shoushui Wei, Zhiqiang He, et al. An open access database for evaluating the algorithms of electrocardiogram rhythm and morphology abnormality detection. *Journal of Medical Imaging and Health Informatics*, 8(7):1368–1373, 2018.

[22] R. Keith Mobley. *An introduction to predictive maintenance*. Butterworth-Heinemann, 2002.

[23] Thyago P. Carvalho, Fabrízzio A.A.M.N. Soares, Roberto Vita, Roberto da P. Francisco, João P. Basto, and Symone G.S. Alcalá. A systematic literature review of machine learning methods applied to predictive maintenance. *Computers and Industrial Engineering*, 137, 2019. doi:10.1016/j.cie.2019.106024.

[24] Yongyi Ran, Xin Zhou, Pengfeng Lin, Yonggang Wen, and Ruilong Deng. A survey of predictive maintenance: Systems, purposes and approaches, 2019. arXiv:1912.07383.

[25] Sule Selcuk. Predictive maintenance, its implementation and latest trends. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, 231:1670–1679, 2017. doi:10.1177/0954405415601640.

[26] Alan V. Oppenheim and Ronald W. Schafer. *Discrete-time signal processing*. Pearson Education India, 1999.

[27] Sophocles J Orfanidis. *Introduction to signal processing*. Prentice-Hall, Inc., 1995.

[28] Donald E Knuth. Big omicron and big omega and big theta. *ACM Sigact News*, 8(2):18–24, 1976.

[29] E. Oran Brigham. *The fast Fourier transform and its applications*. Prentice-Hall signal processing series. Prentice Hall, Englewood Cliffs, N.J, 1988.

[30] A. Graps. An introduction to wavelets. *IEEE Computational Science and Engineering*, 2:50–61, 1995. URL: http://ieeexplore.ieee.org/document/388960/, doi:10.1109/99.388960.

[31] Yonggang Xu, Yunjie Deng, Jiyuan Zhao, Weikang Tian, and Chaoyong Ma. A novel rolling bearing fault diagnosis method based on empirical wavelet transform and spectral trend. *IEEE Transactions on Instrumentation and Measurement*, 69(6):2891–2904, 2019.

[32] Norden E. Huang, Zheng Shen, Steven R. Long, Manli C. Wu, Hsing H. Shih, Quanan Zheng, Nai-Chyuan Yen, Chi Chao Tung, and Henry H. Liu. The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 454, March 1998. URL: https://royalsocietypublishing.org/doi/10.1098/rspa.1998.0193, doi:10.1098/rspa.1998.0193.

[33] Practical Introduction to Multiresolution Analysis - MATLAB & Simulink Example. Accessed on 08-06-2023. URL: https://www.mathworks.com/help/wavelet/ug/practical-introduction-to-multiresolution-analysis.html.

[34] Michael Feldman. Hilbert transform in vibration analysis. *Mechanical systems and signal processing*, 25(3):735–802, 2011.

[35] Norden Eh Huang. *Hilbert-Huang transform and its applications*, volume 16. World Scientific, 2014.

[36] Gang Cheng, Yu-long Cheng, Li-hua Shen, Jin-bo Qiu, and Shuai Zhang. Gear fault identification based on hilbert–huang transform and som neural network. *Measurement*, 46(3):1137–1146, 2013.

[37] Jay Lee, Fangji Wu, Wenyu Zhao, Masoud Ghaffari, Linxia Liao, and David Siegel. Prognostics and health management design for rotary machinery systems—Reviews, methodology and applications. *Mechanical Systems and Signal Processing*, 42:314–334, 2014. doi:10.1016/j.ymssp.2013.06.004.

[38] U Bhardwaj, AP Teixeira, and C Guedes Soares. Reliability prediction of an offshore wind turbine gearbox. *Renewable Energy*, 141:693–706, 2019.

[39] Gustavo Scalabrini Sampaio, Arnaldo Rabello de Aguiar Vallim Filho, Leilton Santos da Silva, and Leandro Augusto da Silva. Prediction of motor failure time using an artificial neural network. *Sensors*, 19(19):4342, 2019.

[40] N Tandon and A Choudhury. A review of vibration and acoustic measurement methods for the detection of defects in rolling element bearings. *Tribology International*, 32(8):469–480, August 1999. doi:10.1016/S0301-679X(99)00077-8.

[41] Ken Mao, DG Chetwynd, and M Millson. A new method for testing polymer gear wear rate and performance. *Polymer Testing*, 82:106323, 2020.

[42] Anurag Choudhary, Deepam Goyal, Sudha Letha Shimi, and Aparna Akula. Condition monitoring and fault diagnosis of induction motors: A review. *Archives of Computational Methods in Engineering*, 26:1221–1238, 2019.

[43] Yaguo Lei, Jing Lin, Ming J Zuo, and Zhengjia He. Condition monitoring and fault diagnosis of planetary gearboxes: A review. *Measurement*, 48:292–305, 2014.

[44] R.B. Randall. *Vibration-based Condition Monitoring: Industrial, Aerospace and Automotive Applications*. Wiley, 2011. URL: https://books.google.pt/books?id=2v4RDQEACAAJ.

[45] Saeed Shokrollahi, Farhad Adel, and Hamid Ahmadian. An investigation into the accelerometer mounting effects on signal transmissibility in modal measurements. *Scientia Iranica*, 24(5):2436–2444, 2017.

[46] Case western reserve university bearing data center 2021, Aug 2021. Accessed on 23-04-2023. URL: https://engineering.case.edu/bearingdatacenter.

[47] Dhiraj Neupane and Jongwon Seok. Bearing Fault Detection and Diagnosis Using Case Western Reserve University Dataset With Deep Learning Approaches: A Review. *IEEE Access*, 2020. doi:10.1109/ACCESS.2020.2990528.

[48] Paderborn university bearing dataset, Aug 2021. Accessed on 23-04-2023. URL: https://mb.uni-paderborn.de/en/kat/main-research/datacenter/bearing-datacenter/data-sets-and-download.

[49] Patrick Nectoux, Rafael Gouriveau, Kamal Medjaher, Emmanuel Ramasso, Brigitte Chebel-Morello, Noureddine Zerhouni, and Christophe Varnier. Pronostia: An experimental platform for bearings accelerated degradation tests. In *IEEE International Conference on Prognostics and Health Management, PHM'12.*, pages 1–8. IEEE Catalog Number: CPF12PHM-CDR, 2012.

[50] Oliver Mey, Willi Neudeck, Andre Schneider, and Olaf Enge-Rosenblatt. Machine Learning-Based Unbalance Detection of a Rotating Shaft Using Vibration Data. In *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1610–1617, Vienna, Austria, September 2020. IEEE. doi:10.1109/ETFA46521.2020.9212000.

[51] Kahiomba Sonia Kiangala and Zenghui Wang. Initiating predictive maintenance for a conveyor motor in a bottling plant using industry 4.0 concepts. *The International Journal of Advanced Manufacturing Technology*, 97(9-12):3251–3271, August 2018. doi:10.1007/s00170-018-2093-8.

[52] N Ahobal et al. Study of vibration characteristics of unbalanced overhanging rotor. In *IOP Conference Series: Materials Science and Engineering*, volume 577, page 012140. IOP Publishing, 2019.

[53] Hongmei Liu, Lianfeng Li, and Jian Ma. Rolling Bearing Fault Diagnosis Based on STFT-Deep Learning and Sound Signals. *Shock and Vibration*, 2016:1–12, 2016. doi:10.1155/2016/6127479.

[54] Yunus Emre Karabacak, Nurhan Gürsel Özmen, and Levent Gümüşel. Intelligent worm gearbox fault diagnosis under various working conditions using vibration, sound and thermal features. *Applied Acoustics*, 186:108463, January 2022. doi:10.1016/j.apacoust.2021.108463.

[55] Meng Ma, Chuang Sun, and Xuefeng Chen. Deep Coupling Autoencoder for Fault Diagnosis With Multimodal Sensory Data. *IEEE Transactions on Industrial Informatics*, 14(3):1137–1145, March 2018. doi:10.1109/TII.2018.2793246.

[56] Eduardo C. Nunes. Anomalous sound detection with machine learning: A systematic review, 2021. arXiv:2102.07820.

[57] Yuma Koizumi, Shoichiro Saito, Hisashi Uematsu, Noboru Harada, and Keisuke Imoto. Toyadmos: A dataset of miniature-machine operating sounds for anomalous sound detection, 2019. arXiv:1908.03299.

[58] Harsh Purohit, Ryo Tanabe, Kenji Ichige, Takashi Endo, Yuki Nikaido, Kaori Suefusa, and Yohei Kawaguchi. Mimii dataset: Sound dataset for malfunctioning industrial machine investigation and inspection, 2019. arXiv:1909.09347.

[59] Noboru Harada, Daisuke Niizumi, Daiki Takeuchi, Yasunori Ohishi, Masahiro Yasuda, and Shoichiro Saito. Toyadmos2: Another dataset of miniature-machine operating sounds for anomalous sound detection under domain shift conditions, 2021. arXiv:2106.02369.

[60] Yaoguang Wang, Yaohao Zheng, Yunxiang Zhang, Yongsheng Xie, Sen Xu, Ying Hu, and Liang He. Unsupervised anomalous sound detection for machine condition monitoring using classification-based methods. *Applied Sciences*, 11(23):11128, 2021.

[61] Paul Primus, Verena Haunschmid, Patrick Praher, and Gerhard Widmer. Anomalous sound detection as a simple binary classification problem with careful selection of proxy outlier examples. *arXiv preprint arXiv:2011.02949*, 2020.

[62] Adam Bondyra, Marek Kołodziejczak, Radosław Kulikowski, and Wojciech Giernacki. An Acoustic Fault Detection and Isolation System for Multirotor UAV. *Energies*, 15(11), June 2022. doi:10.3390/en15113955.

[63] CK Madhusudana, Hemantha Kumar, and S Narendranath. Face milling tool condition monitoring using sound signal. *International Journal of System Assurance Engineering and Management*, 8:1643–1653, 2017.

[64] Yicheng Yu, Ali Safari, Xudong Niu, Bruce Drinkwater, and Kirill V Horoshenkov. Acoustic and ultrasonic techniques for defect detection and condition monitoring in water and sewerage pipes: A review. *Applied Acoustics*, 183:108282, 2021.

[65] Suvi Santa-aho, Antti Nurmikolu, and Minnamari Vippola. Automated ultrasound-based inspection of rails. *International Journal of Railway*, 10(2):21–29, 2017.

[66] Arash Behnia, Hwa Kian Chai, and Tomoki Shiotani. Advanced structural health monitoring of concrete structures with the aid of acoustic emission. *Construction and Building Materials*, 65:282–302, August 2014. doi:10.1016/j.conbuildmat.2014.04.103.

[67] ThomasD. Rossing, editor. *Springer Handbook of Acoustics*. Springer New York, New York, NY, 2007. doi:10.1007/978-0-387-30425-0.

[68] C.U. Grosse, M. Ohtsu, D.G. Aggelis, and T. Shiotani. *Acoustic Emission Testing: Basics for Research – Applications in Engineering*. Springer Tracts in Civil Engineering. Springer International Publishing, 2021. URL: https://books.google.pt/books?id=EYk4EAAAQBAJ.

[69] Tomoki Shiotani. Evaluation of repair effect for deteriorated concrete piers of intake dam using ae activity. In *Acoustic Emission Testing*, volume 13 of *Advanced Materials Research*, pages 175–180. Trans Tech Publications Ltd, 9 2006. doi:10.4028/www.scientific.net/AMR.13-14.175.

[70] Christian U. Grosse and Florian Finck. Quantitative evaluation of fracture processes in concrete using signal-based acoustic emission techniques. *Cement and Concrete Composites*, 28(4):330–336, April 2006. doi:10.1016/j.cemconcomp.2006.02.006.

[71] Manfred Joswig. Pattern recognition for earthquake detection. *Bulletin of the Seismological Society of America*, 80(1):170–186, 1990.

[72] Wahyu Caesarendra, Buyung Kosasih, Anh Kiet Tieu, Hongtao Zhu, Craig A.S. Moodie, and Qiang Zhu. Acoustic emission-based condition monitoring methods: Review and application for low speed slew bearing. *Mechanical Systems and Signal Processing*, 72-73:134–159, May 2016. doi:10.1016/j.ymssp.2015.10.020.

[73] Rende Liu. Condition monitoring of low-speed and heavily loaded rolling element bearing. *Industrial Lubrication and Tribology*, 59(6):297–300, 2007.

[74] Davide Crivelli, Simon Hutt, Alastair Clarke, Pietro Borghesani, Zhongxiao Peng, and Robert Randall. Condition monitoring of rotating machinery with acoustic emission: A british–australian collaboration. In *Asset Intelligence through Integration and Interoperability and Contemporary Vibration Engineering Technologies: Proceedings of the 12th World Congress on Engineering Asset Management and the 13th International Conference on Vibration Engineering and Technology of Machinery*, pages 119–128. Springer, 2019.

[75] Enrique Caso, Alfonso Fernandez-del Rincon, Pablo Garcia, Miguel Iglesias, and Fernando Viadero. Monitoring of misalignment in low speed geared shafts with acoustic emission sensors. *Applied Acoustics*, 159:107092, 2020.

[76] S. Bagavathiappan, B.B. Lahiri, T. Saravanan, John Philip, and T. Jayakumar. Infrared thermography for condition monitoring – A review. *Infrared Physics & Technology*, 60:35–55, September 2013. doi:10.1016/j.infrared.2013.03.006.

[77] A.W. Kandeal, M.R. Elkadeem, Amrit Kumar Thakur, Gamal B. Abdelaziz, Ravishankar Sathyamurthy, A.E. Kabeel, Nuo Yang, and Swellam W. Sharshir. Infrared thermography-based condition monitoring of solar photovoltaic systems: A mini review of recent advances. *Solar Energy*, 223:33–43, July 2021. doi:10.1016/j.solener.2021.05.032.

[78] Olivier Janssens, Raiko Schulz, Viktor Slavkovikj, Kurt Stockman, Mia Loccufier, Rik Van De Walle, and Sofie Van Hoecke. Thermal image based fault diagnosis for rotating machinery. *Infrared Physics & Technology*, 73:78–87, November 2015. doi:10.1016/j.infrared.2015.09.004.

[79] Anam Abid, Muhammad Tahir Khan, and Javaid Iqbal. A review on fault detection and diagnosis techniques: basics and beyond. *Artificial Intelligence Review*, 54:3639–3664, June 2021. doi:10.1007/s10462-020-09934-2.

[80] Reemon Z Haddad and Elias G Strangas. On the accuracy of fault detection and separation in permanent magnet synchronous machines using mcsa/mvsa and lda. *IEEE Transactions on Energy Conversion*, 31(3):924–934, 2016.

[81] Ali Bou Nassif, Manar Abu Talib, Qassim Nasir, and Fatima Mohamad Dakalbab. Machine Learning for Anomaly Detection: A Systematic Review. *IEEE Access*, 9:78658–78700, 2021. URL: https://ieeexplore.ieee.org/document/9439459/, doi:10.1109/ACCESS.2021.3083060.

[82] Erica Raviola and Franco Fiori. A Low-Cost, Small-Size, and Bluetooth-Connected Module to Detect Faults in Rolling Bearings. *Applied Sciences*, 10(16):5645, August 2020. doi:10.3390/app10165645.

[83] Muhammad Masood Tahir, Abdul Qayyum Khan, Naeem Iqbal, Ayyaz Hussain, and Saeed Badshah. Enhancing Fault Classification Accuracy of Ball Bearing Using Central Tendency Based Time Domain Features. *IEEE Access*, 5:72–83, 2017. URL: http://ieeexplore.ieee.org/document/7572105/, doi:10.1109/ACCESS.2016.2608505.

[84] K Balachandar and R Jegadeeshwaran. Friction stir welding tool condition monitoring using vibration signals and random forest algorithm–a machine learning approach. *Materials Today: Proceedings*, 46:1174–1180, 2021.

[85] Candice Bentéjac, Anna Csörgő, and Gonzalo Martínez-Muñoz. A comparative analysis of gradient boosting algorithms. *Artificial Intelligence Review*, 54:1937–1967, March 2021. URL: https://link.springer.com/10.1007/s10462-020-09896-5, doi:10.1007/s10462-020-09896-5.

[86] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[87] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30, 2017.

[88] Anna Veronika Dorogush, Vasily Ershov, and Andrey Gulin. Catboost: gradient boosting with categorical features support. *arXiv preprint arXiv:1810.11363*, 2018.

[89] Dahai Zhang, Liyang Qian, Baijin Mao, Can Huang, Bin Huang, and Yulin Si. A data-driven design for fault detection of wind turbines using random forests and xgboost. *Ieee Access*, 6:21020–21031, 2018.

[90] Zeki Murat Çınar, Abubakar Abdussalam Nuhu, Qasim Zeeshan, Orhan Korhan, Mohammed Asmael, and Babak Safaei. Machine Learning in Predictive Maintenance towards Sustainable Smart Manufacturing in Industry 4.0. *Sustainability*, 12:8211, October 2020. doi:10.3390/su12198211.

[91] Lu Shuang and Li Meng. Bearing fault diagnosis based on pca and svm. In *2007 International conference on mechatronics and automation*, pages 3503–3507. IEEE, 2007.

[92] Miryam Elizabeth Villa-Pérez, Miguel A Alvarez-Carmona, Octavio Loyola-Gonzalez, Miguel Angel Medina-Pérez, Juan Carlos Velazco-Rossell, and Kim-Kwang Raymond Choo. Semi-supervised anomaly detection algorithms: A comparative summary and future research directions. *Knowledge-Based Systems*, 218:106878, 2021.

[93] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation Forest. In *2008 Eighth IEEE International Conference on Data Mining*, pages 413–422. IEEE, December 2008. doi:10.1109/ICDM.2008.17.

[94] Hongfei Wang, Wen Jiang, Xinyang Deng, and Jie Geng. A new method for fault detection of aero-engine based on isolation forest. *Measurement*, November 2021. doi:10.1016/j.measurement.2021.110064.

[95] Shen Yin, Xiangping Zhu, and Chen Jing. Fault detection based on a robust one class support vector machine. *Neurocomputing*, 145:263–268, 2014.

[96] Xinya Wu, Yan Zhang, Changming Cheng, and Zhike Peng. A hybrid classification autoencoder for semi-supervised fault diagnosis in rotating machinery. *Mechanical Systems and Signal Processing*, 149:107327, 2021.

[97] Arden Dertat. Applied Deep Learning - Part 3: Autoencoders, October 2017. URL: https://towardsdatascience.com/applied-deep-learning-part-3-autoencoders-1c083af4d798.

[98] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic robotics*. MIT Press, 2005.

[99] Moslem Azamfar, Jaskaran Singh, Inaki Bravo-Imaz, and Jay Lee. Multisensor data fusion for gearbox fault diagnosis using 2-D convolutional neural network and motor current signature analysis. *Mechanical Systems and Signal Processing*, 144:106861, October 2020. doi:10.1016/j.ymssp.2020.106861.

[100] HT Shi and XT Bai. Model-based uneven loading condition monitoring of full ceramic ball bearings in starved lubrication. *Mechanical Systems and Signal Processing*, 139:106583, 2020.

[101] Tai Guo and Zhongmin Deng. An improved emd method based on the multi-objective optimization and its application to fault feature extraction of rolling bearing. *Applied Acoustics*, 127:46–62, 2017.

[102] Dong Liu, Hongtao Zeng, Zhihuai Xiao, Lihong Peng, and OP Malik. Fault diagnosis of rotor using emd thresholding-based de-noising combined with probabilistic neural network. *Journal of Vibroengineering*, 19(8):5920–5931, 2017.

[103] Roberto M Souza, Erick GS Nascimento, Ubatan A Miranda, Wenisten JD Silva, and Herman A Lepikson. Deep learning for diagnosis and classification of faults in industrial rotating machinery. *Computers & Industrial Engineering*, 153:107060, 2021.

[104] Yaqiong Lv, Wenqin Zhao, Zhiyao Zhao, Weidong Li, and Kam K.H. Ng. Vibration signal-based early fault prognosis: Status quo and applications. *Advanced Engineering Informatics*, 52:101609, April 2022. doi:10.1016/j.aei.2022.101609.

[105] Jacek Dybała and Radosław Zimroz. Rolling bearing diagnosing method based on empirical mode decomposition of machine vibration signal. *Applied Acoustics*, 77:195–203, 2014.

[106] Saeed Nezamivand Chegini, Ahmad Bagheri, and Farid Najafi. Application of a new ewt-based denoising technique in bearing fault diagnosis. *Measurement*, 144:275–297, 2019.

[107] Yaguo Lei and Ming J Zuo. Fault diagnosis of rotating machinery using an improved hht based on eemd and sensitive imfs. *Measurement Science and Technology*, 20(12):125701, 2009.

[108] Tao Liu and Dongxin Lu. The application and development of iot. In *2012 International symposium on information technologies in medicine and education*, volume 2, pages 991–994. IEEE, 2012.

[109] Mustapha Reda Senouci and Abdelhamid Mellouk. *Deploying wireless sensor networks: theory and practice*. Elsevier, 2016.

[110] Sakina Elhadi, Abdelaziz Marzak, Nawal Sael, and Soukaina Merzouk. Comparative Study of IoT Protocols. *SSRN Electronic Journal*, 2018. doi:10.2139/ssrn.3186315.

[111] Jie Ding, Mahyar Nemati, Chathurika Ranaweera, and Jinho Choi. IoT Connectivity Technologies and Applications: A Survey. *IEEE Access*, 8:67646–67673, 2020. doi:10.1109/ACCESS.2020.2985932.

[112] Nurzaman Ahmed, Hafizur Rahman, and Md I Hussain. A comparison of 802.11 ah and 802.15. 4 for iot. *Ict Express*, 2(3):100–102, 2016.

[113] Qahhar Muhammad Qadir, Tarik A. Rashid, Nawzad K. Al-Salihi, Birzo Ismael, Alexander A. Kist, and Zhongwei Zhang. Low Power Wide Area Networks: A Survey of Enabling Technologies, Applications and Interoperability Needs. *IEEE Access*, 6:77454–77473, 2018. doi:10.1109/ACCESS.2018.2883151.

[114] Jin-Shyan Lee, Yu-Wei Su, and Chung-Chou Shen. A comparative study of wireless protocols: Bluetooth, uwb, zigbee, and wi-fi. In *IECON 2007-33rd Annual Conference of the IEEE Industrial Electronics Society*, pages 46–51. Ieee, 2007.

[115] Accessed on 02-07-2023. URL: https://spectraquest.com/simulators/details/mfs/.

[116] S. A. McInerny and Y. Dai. Basic vibration signal processing for bearing fault detection. *IEEE Transactions on Education*, 46(1):149–156, February 2003. doi:10.1109/TE.2002.808234.

[117] Alan S. Morris and Reza Langari. Translational motion, vibration, and shock measurement. In *Measurement and Instrumentation*, pages 569–602. Elsevier, 2021. doi:10.1016/B978-0-12-817141-7.00019-0.

[118] Martin Doseděl and Zdeněk Havránek. Design and performance evaluation of smart vibration sensor for industrial applications with built-in mems accelerometers. In *2018 18th International Conference on Mechatronics-Mechatronika (ME)*, pages 1–8. IEEE, 2018.

[119] Matej Andrejašic. Mems accelerometers. In *University of Ljubljana. Faculty for mathematics and physics, Department of physics, Seminar*, volume 49, 2008.

[120] Analog Devices. *ADXL354/ADXL355 (Rev. B)*, 2020.

[121] NXP Semiconductors. *MMA8451Q, 3-axis, 14-bit/8-bit digital accelerometer*, 2017.

[122] 2019 Adafruit | Mouser. URL: https://pt.mouser.com/ProductDetail/485-2019.

[123] EVAL-ADXL355Z Analog Devices | Mouser. URL: https://pt.mouser.com/ProductDetail/584-EVAL-ADXL355Z.

[124] Siti Aisyah Zawawi, Azrul Azlan Hamzah, Burhanuddin Yeop Majlis, and Faisal Mohd-Yasin. A review of mems capacitive microphones. *Micromachines*, 11(5):484, 2020.

[125] Homin Song, Jongwoong Park, and John S Popovics. Development of an MEMS ultrasonic microphone array system and its application to compressed wavefield imaging of concrete. *Smart Materials and Structures*, 29(10):105011, October 2020. doi:10.1088/1361-665X/ababe5.

[126] GRAS 40PH-10. URL: https://www.ni.com/pt-pt/support/model.gras-40ph-10.html.

[127] knowles. *SPH0645LM4H-1 Rev A Datasheet*, 2017.

[128] 3421 Adafruit | Mouser. URL: https://pt.mouser.com/ProductDetail/485-3421.

[129] Knowles Electronics. *Zero-Height SiSonic TM Microphone*, 2013.

[130] Ultrasonic Mic Board. URL: https://www.veldshop.nl/en/ultrasonic-mic-board.html?source=facebook.

[131] Andrew P. Hill, Peter Prince, Jake L. Snaddon, C. Patrick Doncaster, and Alex Rogers. AudioMoth: A low-cost acoustic device for monitoring biodiversity and the environment. *HardwareX*, 6:e00073, October 2019. doi:10.1016/j.ohx.2019.e00073.

[132] Texas Instruments. *OPAx350 High-Speed, Single-Supply, Rail-to-Rail Operational Amplifiers MicroAmplifier Series*, 2015.

[133] TDK Corporation. *Singing Capacitors (Piezoelectric Effect)*, 2006.

[134] Melexis. *MLX90614 family Single and Dual Zone Infra Red Thermometer in TO-39 Features and Benefits*, 2009. ISBN: 3901090614.

[135] 101020895 Seeed Studio | Mouser. URL: https://pt.mouser.com/ProductDetail/713-101020895.

[136] NXP Semiconductors. *i.MX RT1060 Crossover Processors for Consumer Products*.

[137] NXP Semiconductors. I2c-bus specification and user manual. *NXP Semiconductors*, 4, 2021.

[138] BOB-12009 SparkFun | Mouser. URL: https://pt.mouser.com/ProductDetail/474-BOB-12009.

[139] Matteo Frigo and Steven G. Johnson. The design and implementation of FFTW3. *Proceedings of the IEEE*, 93(2):216–231, 2005. doi:10.1109/JPROC.2004.840301.

[140] Espressif Systems. *ESP8266 Datasheet*.

[141] Ai Thinker. *ESP12E-Datasheet.pdf*.

[142] Espressif Systems. *esp32-c6_datasheet*.

[143] FTDI Chip. *FT232R - Datasheet*.

[144] Mohammad Nasar and Mohammad Abu Kausar. Suitability of influxdb database for iot applications. *International Journal of Innovative Technology and Exploring Engineering*, 8(10):1850–1857, 2019.

[145] Guangyao Zhang, Yi Wang, Xiaomeng Li, Yi Qin, and Baoping Tang. Health indicator based on signal probability distribution measures for machinery condition monitoring. *Mechanical Systems and Signal Processing*, 198:110460, September 2023. doi:10.1016/j.ymssp.2023.110460.