



UNIVERSITY OF
PORTSMOUTH

School of Energy and Electronic Engineering

Name: Boyan Kirilov Mitov

Student Number: 2071735

Project Title: 3 Phase Induction Motor Fault Detection (Student 3)

Course: BEng____Electronic Engineering

Supervisor: Dr. Edward Smart

Academic year: 2023/24

DECLARATION

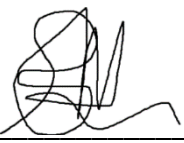
Declaration of Originality and Approval of Research Ethics

Project Title: 3 Phase Induction Motor Fault Detection (Student 3)

Student Name: Boyan Kirilov Mitov

Student Number: 2071735

'I certify that this is my own work, and it has not previously been submitted for any assessed qualification. I certify that School of Engineering research ethics approval has been obtained and the use of material from other sources has been properly and fully acknowledged in the text'.

Signed:  Dated: 9.5.2024

ABSTRACT

Three-phase motors play a crucial role in various industrial, commercial, and residential applications due to their efficiency, reliability, and versatility. Beyond traditional applications, the importance of three-phase motors is also evident in the rapidly growing electric vehicle (EV) industry. However, their continuous and reliable performance is often threatened by various faults that can lead to unexpected downtime, costly repairs, inefficiencies, or in the case of EV's may even cause accidents.

Signal processing is widely recognized as an effective method for detecting faults in three-phase induction motors. The detection of various types of faults has been extensively studied in industrial settings. Building upon existing research, this report aims to provide a comprehensive analysis of different signal processing techniques used for fault detection in three-phase induction motors and the creation of a program which uses few of the mentioned approaches.

The dataset utilized for this project is captured from a 3-phase induction motor with real faults. Leveraging this dataset, a machine learning code was developed, employing two types of machine learning methods. The code learns from healthy motor operation data and utilizes information from faulty motor operation to identify and classify faults. Under certain conditions the code achieved perfect accuracy with balanced error rate of 0%.

Thanks to this code, faults can be detected before they escalate into significant problems, allowing for timely maintenance and preventing costly breakdowns. In the context of electric vehicles (EVs), the ability to detect faults promptly can prevent accidents and ensure the safety of drivers and passengers by addressing potential issues before they compromise vehicle performance or safety.

ACKNOWLEDGEMENTS

I would like to express my heartfelt gratitude to my parents for the incredible opportunity they have given me to study abroad. Their unwavering support, sacrifices, and encouragement have been the driving force behind my journey. I am deeply thankful for their love, guidance, and belief in my abilities.

I also want to extend my sincere appreciation to my supervisor, Dr. Edward Smart, for his invaluable guidance and support throughout this process. His expertise, encouragement, and mentorship have been instrumental in shaping my academic and professional development. I am truly grateful for his patience, wisdom, and dedication to helping me succeed.

Contents

DECLARATION	I
ABSTRACT	II
ACKNOWLEDGEMENTS	III
TABLE OF FIGURES	VI
TABLE OF TABLES	VI
1. INTRODUCTION.....	1
2. LITERATURE REVIEW	3
2.1 3-Phase Motor	3
2.2 Types of motors	4
2.3 Construction	4
2.4 Faults.....	5
2.4.1 Stator Faults	6
2.4.2 Bearing Faults.....	6
2.4.3 Rotor Faults	7
2.4.4 Eccentricity Faults	7
2.5 Fault detection process	8
2.6 Machine Learning Classifiers	9
2.6.1 Supervised Learning	10
2.6.2 Semi-supervised Learning	10
2.6.3 Unsupervised Learning.....	11
2.7 Conclusion.....	11
3. ANALYSIS AND DESIGN	12
3.1 Background analysis	12
3.2 Dataset.....	13
3.3 Signal processing design	16
3.4 Machine Learning Classifier	17
3.5 Software.....	19
3.6 Final Design.....	20
4. IMPLEMENTATION AND SIMULATION.....	21
4.1 Dataset Code.....	21

4.2 Signal Processing Code	23
4.2.1 FFT Code	24
4.2.2 WPT Code	25
4.3 Machine learning code	26
5. TEST AND DISCUSSION	28
5.1 Test	28
5.1.1 FFT	28
5.1.2 WPT	32
5.2 Discussion	33
6. PROJECT EVALUATION/CONCLUSION.....	34
6.1 Evaluation	34
6.2 Future Work.....	34
6.3 Conclusion.....	35
7. REFERENCES	36
LIST OF ACHIEVEMENTS	38
Project Achievements.....	38
Learning Achievements	38

Table of Figures

Figure 2-1 Fleming's left-hand rule.....	3
Figure 2-2 Parts of three-phase motor taken from [4]	5
Figure 3-1 Block Diagram	13
Figure 3-2 3-Phase Induction Motor	13
Figure 3-3 Dataset.....	14
Figure 3-4 Bearing Fault.....	14
Figure 3-5 Eccentricity Fault	15
Figure 3-6 Rotor Fault	15
Figure 3-7 Winding Fault.....	16
Figure 3-8 Labelling of Test Set	17
Figure 3-9 Example Confusion Matrix.....	18
Figure 3-10 Final Design Block Diagram.....	20
Figure 4-1 Data and Metadata Code.....	21
Figure 4-2 Data Table.....	22
Figure 4-3 Heat map chart comparing healthy and bearing fault dataset at 700 RPM, load of 1.5kW, Depth 10 and window length of 12500	23
Figure 4-4 FFT Code	24
Figure 4-5 WPT Code	25
Figure 4-6 ML Code.....	26
Figure 5-1 Decision Score (test 1 healthy 1, 2, FFT)	28
Figure 5-2 Decision Score (test 2 healthy 1, 3, FFT)	29
Figure 5-3 Tests on different Depth and Window length	30
Figure 5-4 Tests on different RPMs and Loads	30
Figure 5-5 Perfect decision Score at 700 RPM using Load 1.5 and training on healthy dataset 1 and 3. FFT approach. ...	31
Figure 5-6 Perfect decision Score at 700 RPM using Load 1.5 and training on healthy dataset 1 and 3. WPT approach. .	32
Figure 5-7 WPT and FFT comparison	33

Table of Tables

Table 3-1 Frequency resolution in Hz at different depths:	17
---	----

1. INTRODUCTION

The first prototype induction motors were developed by Nikola Tesla in the late 19th century. In the subsequent decades, these machines underwent significant improvements, becoming robust, reliable, and efficient. The construction of these motors underwent substantial technological evolution, leading to significant advances in efficiency and power delivery. Furthermore, there has been a reduction in the size and weight of induction motors, enabling greater power within a smaller volume [21].

In today's industrial landscape, three-phase motors are the workhorses that power critical machinery, manufacturing processes, electric vehicles and infrastructure systems.

Induction motors offer several notable advantages. Their high efficiency is a key feature, enabling them to operate effectively across a broad load range. This is a challenge for other types of equipment, such as hydraulic or thermal systems, which experience a rapid drop in efficiency when operated outside their nominal conditions.

Consequently, motors are dominant in the industry, accounting for approximately 55% of energy consumption and utilizing 25% of all generated energy. This dominance is reflected in the market, with the global induction motor market projected to exceed \$58.7 billion by 2030 [21].

EV's are the future of transportation, and ensuring their safety features and reliability is critically important. Detecting motor faults plays a key role in ensuring the overall protection of passengers.

Another example for motors used in manufacturing is the dairy industry, billions of litters of milk are bottled and capped on machines, using 3 phased motors, worth in excess of 1 million GBP, ready for transportation to a supermarket [1].

These motors play a pivotal role in ensuring the seamless operation of countless applications, from conveyor belts and pumps to HVAC systems and production lines.

However, their continuous and reliable performance is often threatened by various faults that can lead to unexpected downtime, costly repairs, and inefficiencies, dairy engineers have conservatively calculated losses of at least 50,000 GBP per day of machine downtime in Europe [2]. To address these challenges, the development and implementation of advanced fault detection systems have become

a necessity, the machines are often operated in hostile conditions and the chemicals used for sterilisation can damage sensors. This fact alone highlights the need for an effective condition monitoring system to quote an industry article “the most effective technique for identifying poor engine conditions: online monitoring.”[21].

This report provides an in-depth exploration of a 3-Phase Motor Fault Detection System, aiming to uncover its significance, underlying principles, methodologies, and real-world applications. By leveraging state-of-the-art technology, these systems not only enhance the operational reliability of three-phase motors but also contribute to increased safety, reduced maintenance costs, and improved overall system efficiency. In the forthcoming sections, we will take a look into dairy filler machines that are powered using inverter driven 1.5kW 3 phase induction motors.

2. Literature Review

2.1 3-Phase Motor

Electric motors rely on the interaction between a current-carrying conductor and a magnetic field. When electric current flows through a conductor, it generates an electromagnetic field around the conductor. The polarity of the resulting magnetic field is determined by the direction of the current. In the case of alternating current (AC), this polarity continually changes. By placing this current-carrying conductor within a magnetic field, it experiences a force known as the Lorentz force. This force is a result of the interaction between the magnetic field and the current, compelling the conductor to move. The magnitude of the force depends directly on the current in the wire and the strength of the magnetic field, the force is greatest when the magnetic field is perpendicular to the conductor [1]. To determine the direction of the force we use Fleming's left-hand rule [20] (*Figure 2-1*).

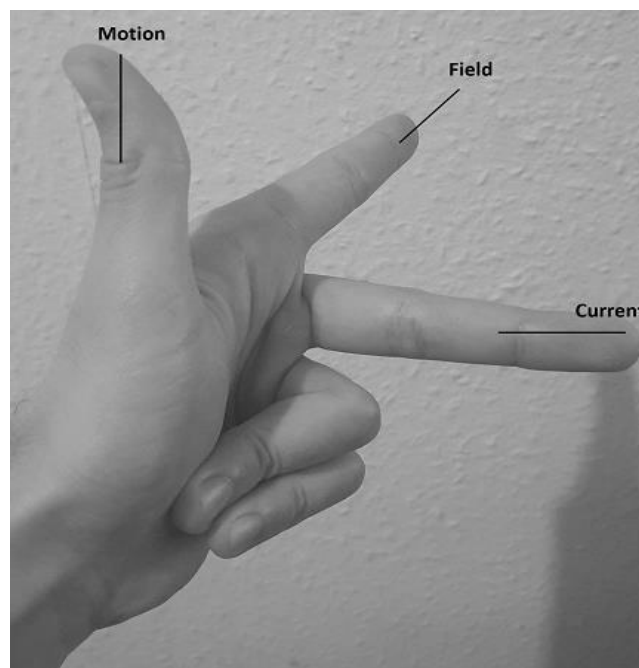


Figure 2-1 Fleming's left-hand rule

The electric motor harnesses this phenomenon to convert electrical energy into mechanical energy, driving the rotation of the motor's components. There are many additional steps in making the motor as efficient as possible. For example, having laminated construction of the motor to reduce the eddy currents losses, flux density, cooling system and more.

2.2 Types of motors

There is a diverse range of electric motors, including conventional motors, brushless DC motors, induction motors, synchronous motors, switched reluctance motors, and stepping motors. Each type has unique characteristics and applications, but all use the same basic principle of converting electrical energy to mechanical energy.

Furthermore, there are currently four classes of squirrel cage induction motors in use: Class A, Class B, Class C, and Class D [22]. Class E and class F are used for practical appliances like blowers and fans due to their reduced starting current.

To summaries their differences:

- Class A: Moderate initial torque, moderate starting current, slip less than 5%, and higher inrush current during starting.
- Class B: Lower starting current, moderate starting torque, slip less than 5%, and commonly used in new installations.
- Class C: Higher starting torque, less initial current, 5% slip at full load, with a double cage rotor design.
- Class D: Initial torque is 275% or larger than the rated torque, lower starting current, higher slip at full load, designed for applications with extremely higher-inertia-type loads.
- Classes E and F (soft-start induction motors) were designed for less starting current and used for loads with less starting torque. These designs have been eliminated, and the four main classes mentioned earlier are more commonly used in practical applications.

2.3 Construction

Induction motors consist of two main parts: stator and rotor (Figure 2-2). The stator is stationary and comprises a set of copper wires (in most cases, three or more) that are wound into coils and placed inside the inner perimeter. These copper wires are coated with a special enamel to prevent electricity from taking the shortest path (short-circuiting) and ensure it travels through the entire length of the wire thus improving the magnetic field. Three separated set of coils are used to produce a rotational electromagnetic field by receiving 3 different phases of AC each rotated by 120 degrees from the previous phase. This arrangement leverages the previously explained electromagnetic induction phenomenon to induce rotation in the motor's rotor.

The rotor is connected to the motor shaft and is responsible for converting electrical energy into mechanical energy. It is generally made up of a cylindrical iron core with conductive bars or coils embedded in it. These conductive elements in the rotor facilitate the flow of current, allowing the rotor to respond to the rotating magnetic field generated by the stator.

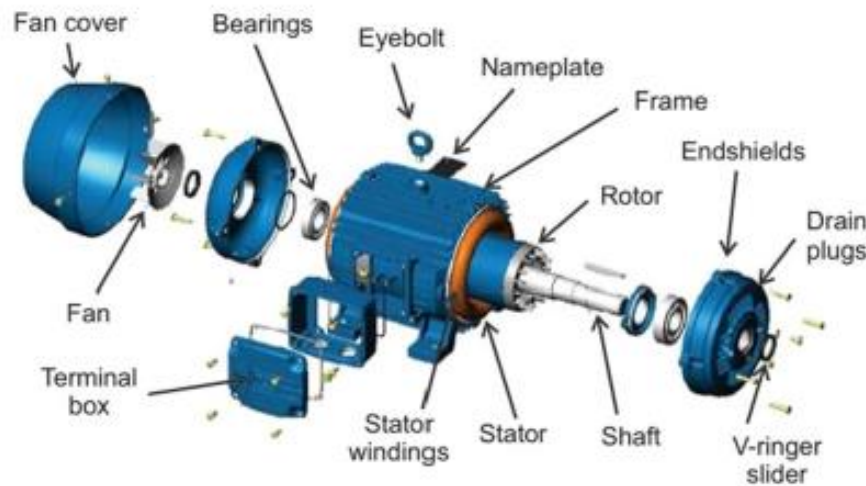


Figure 2-2 Parts of three-phase motor taken from [4]

2.4 Faults

As previously mentioned, motors serve as integral components across various industries, owing to their reliability and versatility. However, these motors are not immune to failure. Several factors contribute to their susceptibility, including challenges in installation, exposure to harsh environmental conditions, and the demands of continuous operation.

One of the primary concerns surrounding induction motors is their increased failure rates. This escalation in failure rates can be caused by variety of factors, such as inadequate installation procedures, suboptimal environmental conditions, and prolonged operational cycles. These issues can exacerbate the wear and tear experienced by the motor components, leading to a higher incidence of faults and defects.

Among the most prevalent issues encountered with induction motors 37% of the time there are stator faults, 41% for bearing problems, 10% rotor defects, and 12% for other [4]. Stator faults, for instance, encompass a range of issues such as insulation degradation, short circuits, and winding

faults. Bearing problems can arise due to factors like lubrication deficiencies, excessive loads, or misalignment. Rotor defects may include issues such as misalignment, eccentricity, or structural weaknesses.

Motor faults are divided into two types: mechanical and electrical. Electrical faults include stator and rotor faults, while mechanical faults consist of eccentricity, load, and bearing faults.

2.4.1 Stator Faults

Stator winding faults pose a significant concern, with insulation being the key vulnerability within the winding structure. The insulation, comprising of mica paper layers on a fiberglass base, saturated with synthetic resin, is crucial for protecting the winding. However, it is prone to various faults. Short-circuiting of a few turns within the winding, known as inter-turn faults, is a major issue. These faults create a substantial circulating current within the coil, leading to its deterioration. Inter-turn faults can be highly destructive and, if left undiagnosed, can escalate to more severe failures such as coil-to-coil or phase-to-phase short circuits, and ultimately phase-to-ground faults.

Insulation failure is particularly concerning as it can result in the burning and localized melting of the winding and core materials. Inter-turn insulation failures often stem from damage to the insulation, typically in the core area, and frequently occur within the first or second coil.

Failure to diagnose stator winding faults early on can have dire consequences, potentially leading to catastrophic motor failure. These hidden faults can disrupt production processes, necessitating unscheduled shutdowns and incurring significant costs in terms of maintenance, lost production time, and material wastage.

Frequently, there are no early warnings, and online monitoring systems may struggle to detect faults promptly. However, in the case of shorted turns within the stator winding, it is feasible to identify their frequency components [2].

2.4.2 Bearing Faults

Bearing failures are a progressive concern in induction motors and stand as the primary cause of catastrophic failures. Dust and corrosion represent the most common culprits behind bearing malfunctions, particularly in the harsh environments where induction motors operate. Foreign particles, water, acidic substances, and humidity contribute significantly to bearing degradation. Poor

installation practices, whether exerting excessive force on the shaft or within the housing, frequently lead to physical damage such as brinelling, accelerating premature failure. Additionally, defective installation may result in misalignment, further exacerbating issues. Broken bearings cause mechanical displacement, altering the motor's air gap and destabilizing rotational eccentricities.

Moreover, bearing failure can affect shaft currents, compounding the complexity of the issue and further disrupting motor operation. Signs and symptoms of bearing failure include mechanical vibration, rising temperatures, decreasing average torque, shifts in torque and speed, increased losses, reduced efficiency, and asymmetry in line current, air gap flux, and voltages.

Cage defects, ball bearing defects, inner and outer race defects generate frequencies that can be calculated using equations based on various parameters. These parameters include the motor shaft rotational frequency in hertz, the diameter of the ball bearing, the diameter of the bearing itself, the contact angle between the ball bearings and the cage, and the number of balls. Detailed equations and their derivation are available in [5].

2.4.3 Rotor Faults

Thermal and mechanical stresses pose significant concerns for the rotor, leading to faults, especially under harsh thermal conditions. Signs of rotor faults include current pulsation, speed irregularities, and stray leakage flux. Diagnosing rotor winding faults is challenging due to the absence of an essential electrical link and difficulty in detecting induced low-frequency currents.

If there's any asymmetry in either the supply or the stator winding impedances, it results in the appearance of a backward rotating field. This phenomenon extends to the rotor winding, causing the induced electromagnetic force frequency to manifest at slip frequency rather than the supply frequency. Consequently, the frequencies at which signs of broken rotor bar faults manifest in the stator current can be calculated.

2.4.4 Eccentricity Faults

In induction motors, rotor and stator eccentricity ranks among the major faults due to its significant contribution to motor failure. Eccentricity leads to unbalanced radial forces, which can result in stator and rotor rubbing, causing damage to both components. Sources of eccentricity include an elliptical inner cross-section of the stator, misalignment between the stator and rotor during commissioning, misalignment between the load and shaft, unbalanced loads, and mechanical resonance.

Air gap eccentricity in induction motors occurs when there is an unequal gap between the stator and rotor. This can manifest in two types: static and dynamic. Static eccentricity arises when the axis of rotation is displaced, leading to a constant minimal air gap length. This displacement may result from incorrect rotor positioning during manufacture or stator ovality. As a consequence, the field distribution in the air gap becomes asymmetrical, generating a radial force known as unbalanced magnetic pull, which acts towards the minimal air gap. Dynamic eccentricity, on the other hand, occurs when the rotor fails to rotate on its axis, causing variation in the minimal air gap length with rotor position.

2.5 Fault detection process

Induction motor failure can be associated with the breakdown of critical components such as the stator, rotor, and bearings. When these components experience faults, various symptoms may arise, indicating potential issues within the motor system. These symptoms include unbalanced air-gap and flux, increased torque and speed variation, decreased average torque, excessive heating, excessive mechanical vibrations, reduced efficiency, increased losses, as well as deviation and asymmetry in currents and voltages. It is crucial to address these symptoms promptly to prevent further damage and ensure the smooth operation of the induction motor [6].

A number of condition monitoring techniques have been developed to detect these fault symptoms of induction motors: magnetic flux (or axial leakage flux), including air gap torque monitoring, acoustic noise measurement, thermal monitoring, partial discharge measurement, instantaneous angular speed, instantaneous power, surge testing, chemical analysis (lubricating oil debris; cooling gas), vibration monitoring, and current monitoring. However, most of the methods are expensive, complex, invasive, and/or not capable of providing rich information about motor conditions. Besides, most of these methods can easily identify a particular induction motor fault, meaning other faults cannot be detected using the same methods. Nowadays, vibration and current monitoring are the most preferable techniques for induction motors. These techniques are known for being non-intrusive, reliable, and cost-effective. They offer precise signal analysis to indicate the current condition of the machine. The data obtained is easily processed for further analysis. They have the capability to identify and differentiate between different mechanical and electrical faults effectively. Moreover, the ability to acquire motor current and vibration data online enables online fault detection [6].

Current measurement alone is insufficient for effectively detecting faults in induction motors. Signal processing techniques are required for fault detection. There are various signal processing approaches for analysing current measurements, including:

1. **Fast Fourier Transform (FFT):** FFT is a method that transforms time-domain signals into their frequency-domain representation. Applying FFT to current signals allows us to identify frequency components associated with faults. For example, irregularities in the FFT spectrum may indicate rotor bar defects or bearing faults [11].
2. **Motor Current Signature Analysis (MCSA):** MCSA involves analysing the current waveform of the motor to detect faults. By examining the unique current signature of the motor, MCSA can identify various faults, including rotor bar defects, broken rotor bars, and electrical supply issues [13].
3. **Wavelet Transform:** Wavelet transform is a technique used to analyse signals in both time and frequency domains simultaneously. It is particularly useful for detecting transient events and localized changes in the current waveform. Wavelet analysis can help identify faults such as short circuits, inter-turn faults, and mechanical impacts [11].
4. **Envelope Analysis:** Envelope analysis focuses on extracting the envelope of the current signal. This technique is effective for detecting modulations in the signal caused by faults. For instance, bearing defects often cause amplitude modulations in the current signal due to the presence of rolling elements [12].
5. **Park's Vector Approach:** Park's vector approach involves transforming the current and voltage vectors into a rotating reference frame. This method is commonly used for diagnosing rotor-related faults such as broken rotor bars and eccentricity [11].

After signal processing, typically a machine learning process is employed to detect faults.

2.6 Machine Learning Classifiers

Machine learning involves using algorithms that analyse input data to make predictions within a certain range. These algorithms learn and improve their performance over time as they receive new data.

There are four main types of machine learning algorithms: supervised, semi-supervised, unsupervised, and reinforcement learning [8].

2.6.1 Supervised Learning

Supervised learning involves teaching a machine through examples. The algorithm is provided with a known dataset containing inputs and desired outputs. It learns from this data, identifying patterns and making predictions. The main tasks under supervised learning are:

- Classification: Categorizing data into different classes. For example, classifying emails as "spam" or "not spam";
- Regression: Estimating relationships between variables to predict outcomes, commonly used in forecasting and predictive modelling.

For example, the Naïve Bayes Classifier Algorithm [14]: This algorithm is a common method in supervised learning. It predicts categories or classes of data based on given features. Naïve Bayes treats each feature as independent from others, making it straightforward yet effective in classification tasks. Despite its simplicity, it often performs well compared to more complex algorithms.

Another example is Artificial Neural Networks (ANN) [18]: These models are inspired by the biological structure of the brain. They consist of interconnected processing units arranged in layers, learning from examples and experiences. ANN's strength lies in its ability to model complex relationships in data, making it particularly useful in various applications.

2.6.2 Semi-supervised Learning

Semi-supervised learning uses both labelled and unlabelled data. Labelled data has tags for the algorithm to understand, while unlabelled data does not. By combining both types of data, the algorithm learns to label unlabelled data.

An instance of semi-supervised learning is the one-class Support Vector Machine (SVM), distinct from the two-class SVM, which is supervised as it requires two classes (healthy and faulty). [16]: SVMs are a type of machine learning algorithm used for classification and regression tasks. They create a model based on provided training data to separate different categories within the data, thereby allowing them to be supervised as well. This model can then classify new data points into one of these categories. The One-Class Support Vector Machine is a unique version of the Support Vector Machine (SVM) tailored for outlier, anomaly, or novelty detection. Its main goal is to identify instances that deviate significantly from the norm. Unlike traditional Machine Learning models, which are typically

used for binary or multiclass classification tasks, the one-class SVM focuses on detecting outliers or novelties within the dataset.

2.6.3 Unsupervised Learning

Unsupervised learning involves exploring data to find patterns without any explicit guidance. The algorithm identifies structures and relationships within the data. The main tasks under unsupervised learning are:

- Clustering: Grouping similar data together based on certain criteria;
- Dimension Reduction: Simplifying data by reducing the number of variables while retaining important information.

An example of unsupervised learning is the K Means Clustering Algorithm [17]: K Means Clustering is an unsupervised learning algorithm used to categorize unlabelled data. It groups data points based on similarities, where the number of groups (K) is predefined. The algorithm iteratively assigns data points to groups based on their features.

2.7 Conclusion

I explored various condition monitoring techniques for detecting faults in induction motors, emphasizing the importance of timely fault detection to prevent further damage and ensure smooth operation. Among these techniques, vibration and current monitoring emerged as the most preferable due to their non-intrusive, reliable, and cost-effective nature, offering precise signal analysis and the ability to differentiate between different faults effectively.

However, I acknowledged that current measurement alone is insufficient for effective fault detection in induction motors. Therefore, I chose to make use of signal processing techniques to enhance fault detection. Specifically, I selected Fast Fourier Transform (FFT) and Wavelet Transform (WPT). FFT is chosen due to its ease of computation and ability to process large amounts of data quickly. It provides excellent frequency resolution, enabling precise identification of frequency components in the signal. Additionally, FFT is widely adopted and supported in software libraries and tools. It offers a clear visualization of the frequency spectrum, facilitating the identification of dominant frequencies and harmonic content. Furthermore, FFT assumes linearity and time-invariance in the system under analysis, which is often reasonable for many motor systems. It has a few disadvantages, such as limited time resolution, which may not capture transient events or changes in the signal occurring

over short time intervals. Additionally, there's the issue of aliasing when the sampling rate isn't high enough to accurately represent high-frequency components. Moreover, when applying FFT to finite-length signals, windowing is typically used to mitigate spectral leakage.

WPT on the other hand offers flexibility in signal decomposition, allowing for a more detailed analysis of frequency components compared to FFT [7]. Provides excellent time-frequency localization, allowing for the analysis of transient events and localized changes in the signal and enables multi-resolution analysis, allowing for the examination of different frequency bands with varying resolutions. The wavelet approach can be computationally complex, potentially increasing processing time and resource requirements. Compared to FFT, WPT is less widely adopted, which may result in fewer available software tools and libraries for implementation. Additionally, interpreting the results of WPT can be more challenging.

After signal processing, I will utilize a machine learning approach for fault detection. With the dataset provided for this project, which I will present later, we have examples of each fault. This suggests that we could create a two-class or even a five-class ML classifier to output "true/false" or "Healthy/Bearing/Winding/Eccentricity/Rotor" respectively. However, in industrial scenarios, it's uncommon to have a balanced dataset with plenty of examples of each fault. Instead, there is usually an abundance of healthy data with either no faults or very few faults. Therefore, I chose the One-Class Support Vector Machine (SVM).

Additionally, this approach is selected because in modern times, when a fault occurs in a motor, the entire motor is often replaced instead of just the faulty part. This is because it's more cost-effective and less time-consuming. Thus, making the report more applicable to industry.

3. ANALYSIS AND DESIGN

3.1 Background analysis

Raw signals can be challenging to interpret, and on their own, they often do not provide sufficient information to determine if a system is healthy or faulty. This is why I employ signal processing techniques to analyse the characteristics of the signal. Signal processing allows us to examine the

signal's properties in the time domain, frequency domain, and time-frequency domain, enabling us to extract fault indicator features.

These extracted features serve as the basis for creating training and testing datasets. The machine learning classifier learns from these datasets to automatically detect faults in the system. By training on labelled data, the classifier can recognize patterns indicative of faults and distinguish them from normal operation thus automatically detecting faults (Figure 3-1).

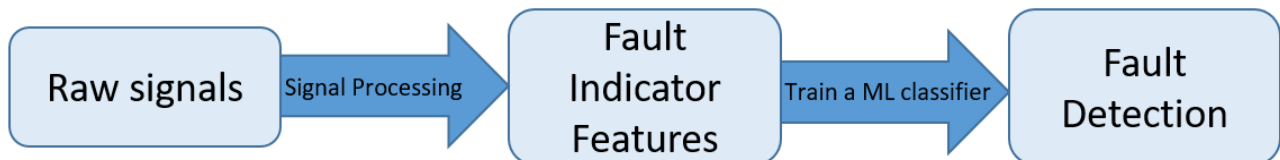


Figure 3-1 Block Diagram

3.2 Dataset

The dataset is collected from an actual three-phase motor used in the industry in China (Figure 3-2). The motor used was a three-phase asynchronous induction motor with a power output of 1.5 kW, a voltage of 380 V, a current of 3.55 A, a speed of 1496 RPM, an efficiency of 78.5% and designed to operate off a mains supply of 50 Hz. Motor speed was changed via an inverter. A current sensor is applied on one of the phases to capture the signals and save them to a computer.

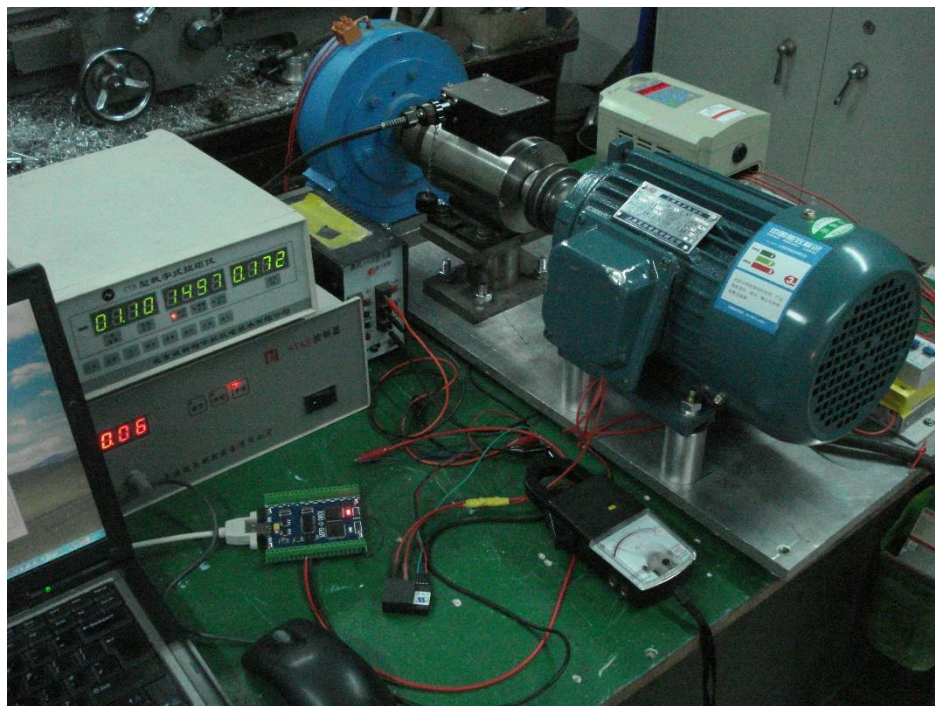


Figure 3-2 3-Phase Induction Motor

The dataset consists of three different speeds: 700, 1000, and 1500 RPM. At each speed, we have three different loads: 0, 0.75, and 1.5 KW. For each load, five different states were recorded: healthy, bearing fault, winding fault, eccentricity fault, and rotor fault, each recorded three times (except for the winding fault which was only studied at 1500 RPM) (see Figure 3-3). Making a total of 117 sets of data.

RPM \ Load	700	1000	1500
0	3x(H,B,E,R)	3x(H,B,E,R)	3x(H,B,W,E,R)
0,75	3x(H,B,E,R)	3x(H,B,E,R)	3x(H,B,W,E,R)
1,5	3x(H,B,E,R)	3x(H,B,E,R)	3x(H,B,W,E,R)

Figure 3-3 Dataset

To simulate real faults, several adjustments were made:

- For the bearing fault, damage was induced using a drill to affect the outer and inner races, ball bearings, and cage (Figure 3-4).

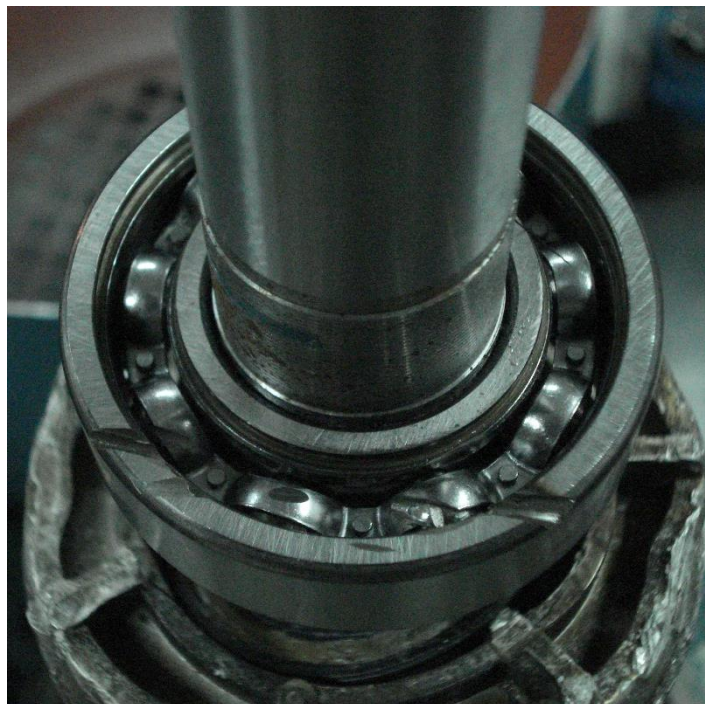


Figure 3-4 Bearing Fault

To create eccentricity, a metal bolt was attached to the motor shaft to induce dynamic eccentricity (Figure 3-5).

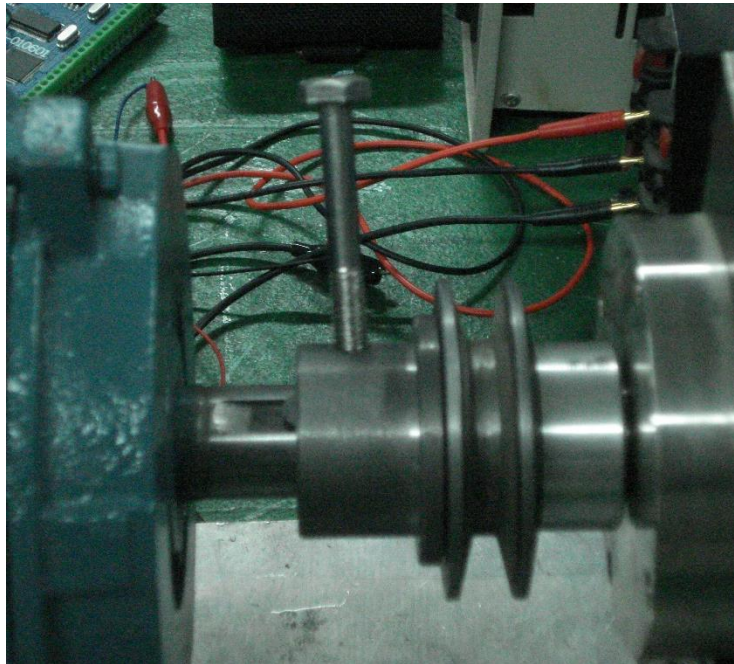


Figure 3-5 Eccentricity Fault

- A rotor fault was introduced by damaging one rotor bar with a drill (Figure 3-6).

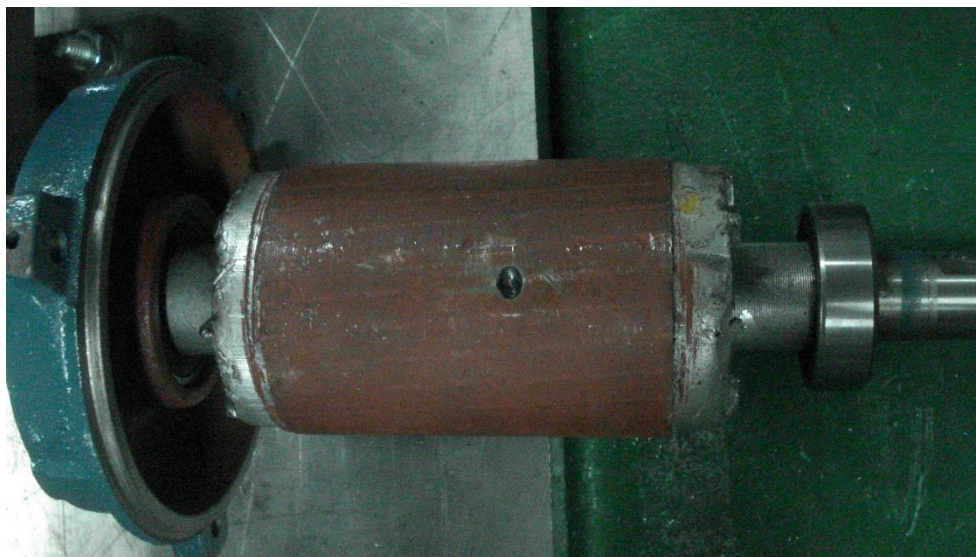


Figure 3-6 Rotor Fault

- The winding fault was simulated by disconnecting one of the motor current phases. It's worth noting that because the speed was adjusted using an inverter to generate all the data, the winding fault only has data available at 1500 RPM (Figure 3-7).

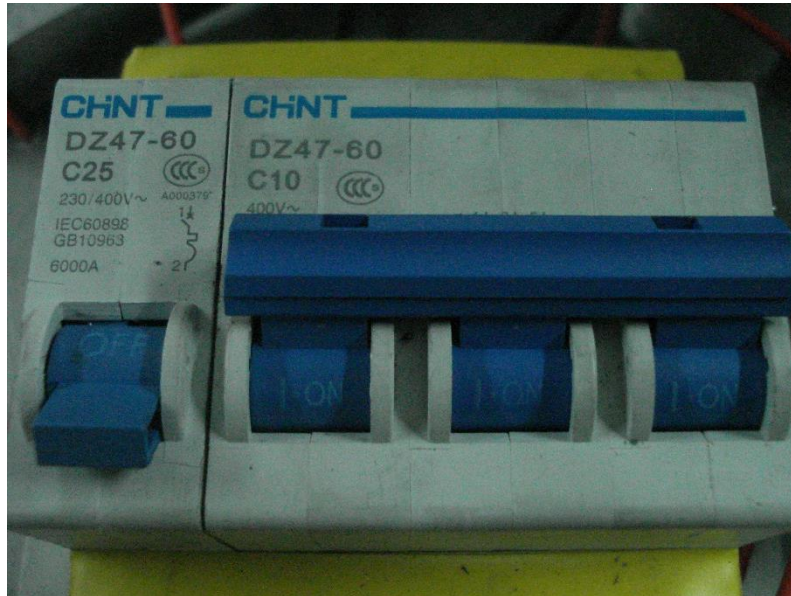


Figure 3-7 Winding Fault

3.3 Signal processing design

Applying signal processing to the whole signal could mean that certain fault signs are missed or minimised within the signal. So, the signal is split into segments and windowing is applied. Windowing is a technique used to reduce the amplitude of discontinuities at the boundaries of each finite sequence acquired by the digitizer. It involves multiplying the time record by a finite-length window with an amplitude that smoothly decreases towards zero at the edges. This gradual decrease in amplitude ensures that the endpoints of the waveform meet, resulting in a continuous waveform without sharp transitions. The 117 signal samples were analysed using a Hanning window of length 5000. Each signal was sampled at 5 kHz and processed using FFT and WPT (daubechies 10 mother wavelet). However, multiple articles stated that almost all kinds of wavelets (e.g. Haar, Morlet, Symlets) provide acceptable outcomes pertaining to the fault diagnosis process [9, 10]. Depth ranging from 1 to 7 is applied to the signal processing approaches. “Depth” refers to the number of times a high pass and a low pass filter was applied to the signal during the WPT process. The greater the depth, the finer the resolution of the decomposed signal. The FFT was computed such that the frequency resolution of the analysed signal matched that of WPT at different depths, so that the two

approaches could be properly compared. A larger number of data points (i.e., higher depth) results in higher frequency resolution, allowing for more precise distinction between frequencies. Conversely, a shorter signal leads to lower frequency resolution. The formula for frequency resolution is shown here:

$$FR = 0.5 * FS / 2^{\text{Depth}}.$$

Here FR stand for frequency resolution and FS is the frequency sample. We have a sample rate of 5000(Hz) and if we use depth from 1 to 10, we get the following table (Table 3-1) of frequency resolutions (Hz):

Table 3-1 Frequency resolution in Hz at different depths:

1250	625	312.5	156.25	78.125	39.0625	19.5312	9.7656	4.8828	2.4414
------	-----	-------	--------	--------	---------	---------	--------	--------	--------

3.4 Machine Learning Classifier

After applying signal processing, 67% of the healthy data is used as training data for the single-class machine learning model or one-class-SVM (support vector machine). The rest of the healthy data and all the faulty data then is used for testing the ML's ability to detect faults. An ideal output is illustrated in (Figure 3-8). In this example, 10 healthy data sets were provided, followed by 10 faulty data sets. The ideal machine learning model would label all the first 10 data sets as healthy (Decision score greater than zero) and the rest as faulty (less than zero).

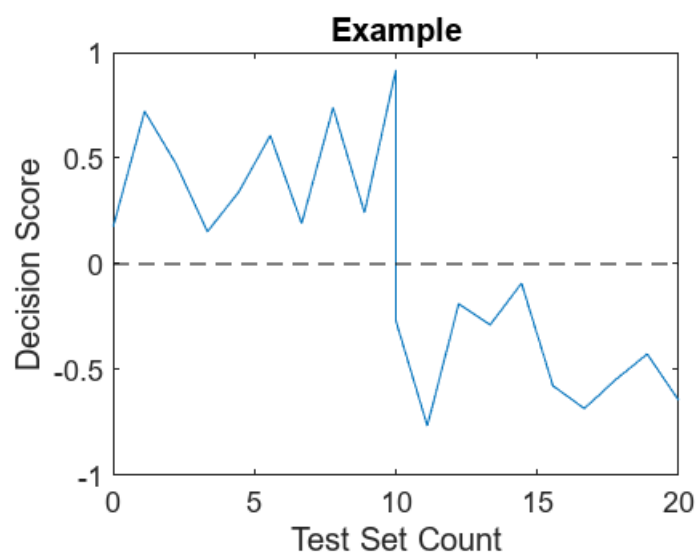


Figure 3-8 Labelling of Test Set

I have installed the LIBSVM library[14] in Matlab. LIBSVM is a software package for support vector classification. To train the system, three things are needed: the labels of the training data (which will be an array of ones with the same size as the training data, as we are training on only healthy data), the training data itself, and a string that sets the operational settings of the SVM code.

This string uses a character and a number to specify the desired setting. For selecting the type of SVM, we use “-s” followed by a number representing the SVM we want. I chose 2 for a single-class SVM, but there are five classes available: (0 -- C-SVC, 1 -- nu-SVC, 2 -- one-class SVM, 3 -- epsilon-SVR, 4 -- nu-SVR).

To choose the kernel type, we use “-t” followed by a number. The options are (0 – linear, 1 – polynomial, 2 -- radial basis function, 3 – sigmoid), each using a different equation. For this project, I chose the radial basis function (RBF) by setting the number to 2, as it employs an exponential equation and is likely to produce the best results. This is because the exponential function can be written as an infinite polynomial. The linear kernel is a polynomial of degree 1 (a straight line) and the polynomial kernel is a finite version of the exponential RBF kernel.

Since I opted for the single-class SVM, I need to set the nu parameter and the gamma. For this, we use the characters “-n” and “-g” respectively. The nu parameter acts as both a lower bound for the number of support vectors and an upper bound for the number of misclassified samples. Because we have healthy data we can trust, I set it to 0.01, which is quite low. The gamma in the kernel function is set to 0.1.

To calculate the accuracy of the machine learning classifier, we need a confusion matrix (Figure 3-9).

		Predicted	
		Healthy	Faulty
Actual	Healthy	384 TP	35 FP
	Faulty	504 FN	3165 TN

Figure 3-9 Example Confusion Matrix

A confusion matrix is a table used to evaluate the performance of a classification algorithm. It includes four categories: True Positive (healthy data correctly labeled as healthy), False Positive (healthy data incorrectly labeled as faulty), False Negative (faulty data incorrectly labeled as healthy), and True Negative (faulty data correctly labeled as faulty).

To determine the balanced error rate (BER), we use the following formula: $BER = ((FP/(FP+TP) + FN/(FN+TN)))/2$ which gives us the average error across the healthy and faulty classes. This formula calculates the percentage of error in the classifier's predictions.

3.5 Software

The program is written in MATLAB [19], which offers a wide range of powerful tools and functions specifically designed for signal analysis and processing tasks. MATLAB provides an extensive set of built-in functions and libraries for various signal processing techniques, including filtering, spectral analysis, waveform generation, and more.

Additionally, MATLAB's intuitive programming environment allows for easy implementation and experimentation with different algorithms and methods. It also provides interactive visualization tools that aid in understanding and interpreting signal data.

I added the Signal Processing Toolbox, which provides functions and apps to manage, analyze, preprocess, and extract features from uniformly and nonuniformly sampled signals. This toolbox enables the use of the Signal Analyzer app for visualizing and processing signals simultaneously in time, frequency, and time-frequency domains. The Signal Labeler app is another interactive tool that enables signal labeling for analysis or for use in machine learning and deep learning applications.

Another toolbox I included is the Wavelet Toolbox, which offers apps and functions for time-frequency analysis of signals and multiscale analysis of images. This toolbox allows denoising and data compression, as well as anomaly detection, change-point detection, and transient analysis within signals. It supports data-centric artificial intelligence workflows by providing time-frequency transforms and automated feature extraction methods, including scattering transforms, continuous wavelet transforms (scalograms), Wigner-Ville distribution, and empirical mode decomposition. Additionally, it allows extraction of edges and oriented features from images using wavelet, wavelet packet, and shearlet transforms.

Finally, I added the LIBSVM library[14] instead of the Statistics and Machine Learning Toolbox for one-class implementation. LIBSVM's core code is in C++, which makes it extremely fast to execute and train each part of the SVM values we provide.

3.6 Final Design

The final design of the program (Figure 3-10) includes the ability to apply windowing to the given raw data, which is the current data provided in section 3.2. Then, signal processing is applied to each

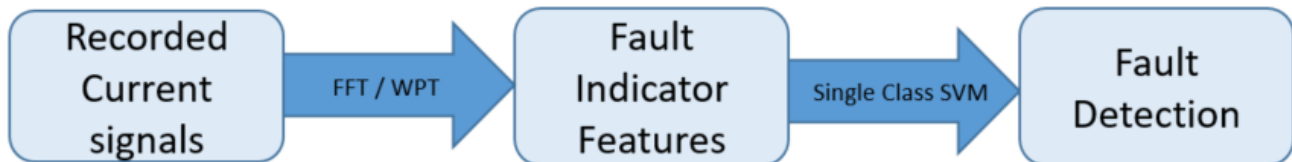


Figure 3-10 Final Design Block Diagram

segment, as chosen and explained in section 3.3, using either wavelet transform (WPT) or fast Fourier transform (FFT). This process generates fault indicator features, where for WPT, it's the percentage of signal energy in each frequency band, and for FFT, and it's the signal amplitude in that frequency band. Subsequently, a single-class SVM, selected in chapter 3.4, is trained on some of the labelled healthy data and tested using the remaining provided data. The program produces a table of labelled data, indicating whether each sample is healthy or faulty. Additionally, machine learning metrics such as BER and confusion matrix, as mentioned in section 3.4, are used to evaluate the results.

4. IMPLEMENTATION AND SIMULATION

4.1 Dataset Code

For the recorded raw data (current signals) of the 3-phase motor. I began by created a string called "directory" to hold the path to the raw data files (Figure 4-1) and then I created structure array called files that uses the "dir" function in MATLAB to list all files from the directory. After initializing the "data" variable as a cell array to match the number of files and setting up variables to store training and testing data, I defined three variables as doubles: one for the sample rate (set to 5000), one for the window length, and one for the depth. I used different values for testing purposes. Then, I created a string to hold the name of the approach, which could be either "WPT" or "FFT".

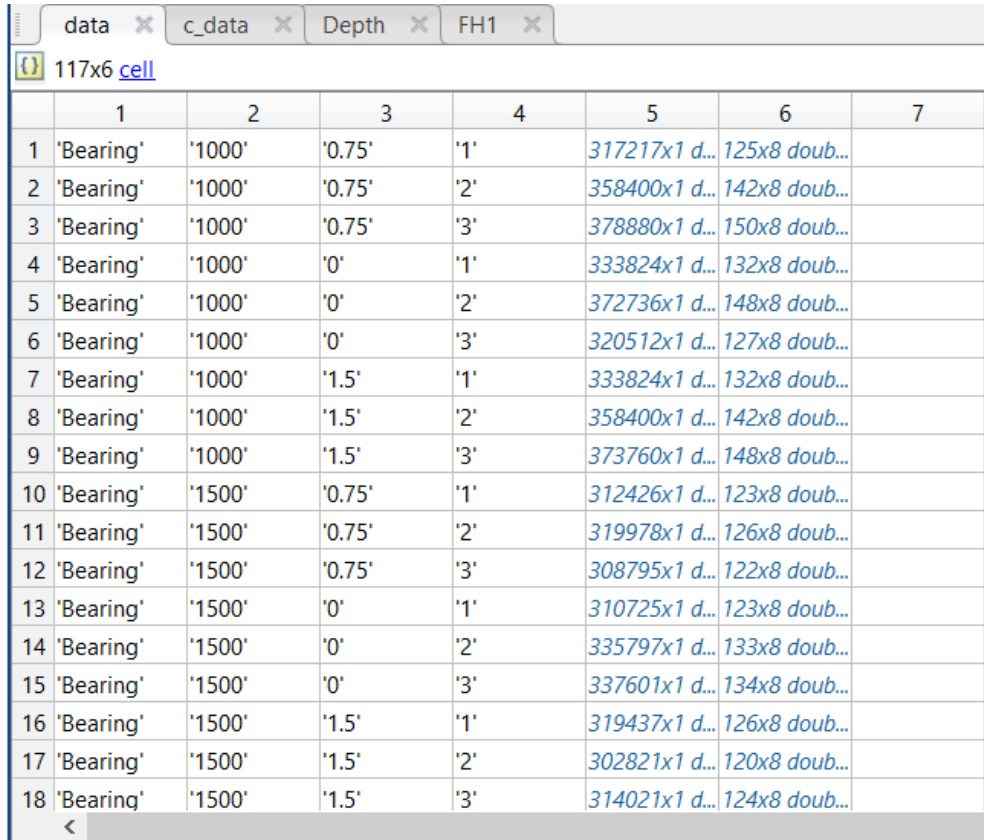
```
7      %Making an archive of all given files
8      directory = 'C:\Users\Kiril Mitov\Desktop\MOTOR\**/*.txt';
9      files = dir(directory); % Using the dir function to list files
10     data = cell(length(files), 5); %making data a cell array
11     t_data = []; %training data
12     f_data = []; %faulty test data
13     h_data = []; %healthy test data
14     Depth = 1;
15
16     Sample_rate = 5000;
17     WinLength = 5000;
18     Aproach = "fft";
19     t=1; % temporary value
20     % Loop through the files and do record each file
21     for i = 1:length(files)
22         %for i = length(files)-30:length(files)
23         if ~files(i).isdir % Ignore directories
24             disp(files(i).name); % Access file name using files(i).name
25             FileName = strcat(files(i).folder, '\', files(i).name);
26             c_data = readmatrix(FileName);
27             % Read the data from the file
28             temp = strsplit(replace(files(i).name, '.txt', ''), '_');
29             data{i,1} = temp(1);
30             data{i,2} = temp(2);
31             data{i,3} = temp(3);
32             data{i,4} = temp(4);
33             data{i,5} = c_data;
34             data{i,6} = ans_fft(data{i,5}, Depth, Sample_rate, Aproach, WinLength);
```

Figure 4-1 Data and Metadata Code

To record each file, I used a for loop that counts from 1 to 117, which is the length of the dataset. Within this loop, I checked if the file is not a directory using the condition "`~files(i).isdir`" and construct the full file path using "`strcat`". Initially, I used "`fopen`" for each txt file, but I found that the "`readmatrix`" function in MATLAB is much faster to compute and doesn't require me to close the

folder afterwards, so I used it instead. Then I extract information from the file name using “strsplit” to get the type of fault, RPM, load, and test number.

I decided to store the data in a cell called "data", (Figure 4-2) where the first column holds the type of fault, the second column holds the RPM, the third column holds the load, the fourth column holds



	1	2	3	4	5	6	7
1	'Bearing'	'1000'	'0.75'	'1'	317217x1 d...	125x8 doub...	
2	'Bearing'	'1000'	'0.75'	'2'	358400x1 d...	142x8 doub...	
3	'Bearing'	'1000'	'0.75'	'3'	378880x1 d...	150x8 doub...	
4	'Bearing'	'1000'	'0'	'1'	333824x1 d...	132x8 doub...	
5	'Bearing'	'1000'	'0'	'2'	372736x1 d...	148x8 doub...	
6	'Bearing'	'1000'	'0'	'3'	320512x1 d...	127x8 doub...	
7	'Bearing'	'1000'	'1.5'	'1'	333824x1 d...	132x8 doub...	
8	'Bearing'	'1000'	'1.5'	'2'	358400x1 d...	142x8 doub...	
9	'Bearing'	'1000'	'1.5'	'3'	373760x1 d...	148x8 doub...	
10	'Bearing'	'1500'	'0.75'	'1'	312426x1 d...	123x8 doub...	
11	'Bearing'	'1500'	'0.75'	'2'	319978x1 d...	126x8 doub...	
12	'Bearing'	'1500'	'0.75'	'3'	308795x1 d...	122x8 doub...	
13	'Bearing'	'1500'	'0'	'1'	310725x1 d...	123x8 doub...	
14	'Bearing'	'1500'	'0'	'2'	335797x1 d...	133x8 doub...	
15	'Bearing'	'1500'	'0'	'3'	337601x1 d...	134x8 doub...	
16	'Bearing'	'1500'	'1.5'	'1'	319437x1 d...	126x8 doub...	
17	'Bearing'	'1500'	'1.5'	'2'	302821x1 d...	120x8 doub...	
18	'Bearing'	'1500'	'1.5'	'3'	314021x1 d...	124x8 doub...	

Figure 4-2 Data Table

the test number, the fifth column holds the raw signals, and the sixth column holds the processed signal values.

4.2 Signal Processing Code

To use all of the above, I created a function “ans_fft” that takes the raw signals, the sample rate, the approach string, and the window length. It applies windowing using the "hanning" function in MATLAB to the window size, then calculates the resolution using the formula $FR = 0.5 * FS / 2^{\text{Depth}}$. Afterwards, it either applies FFT or WPT depending on the selected approach. I compared healthy and faulty datasets using both FFT and WPT and plotted them on a heat map chart (Figure 4-3). It is evident that WPT has a higher resolution because it is using both frequency and time domain, making it better for displaying faults.

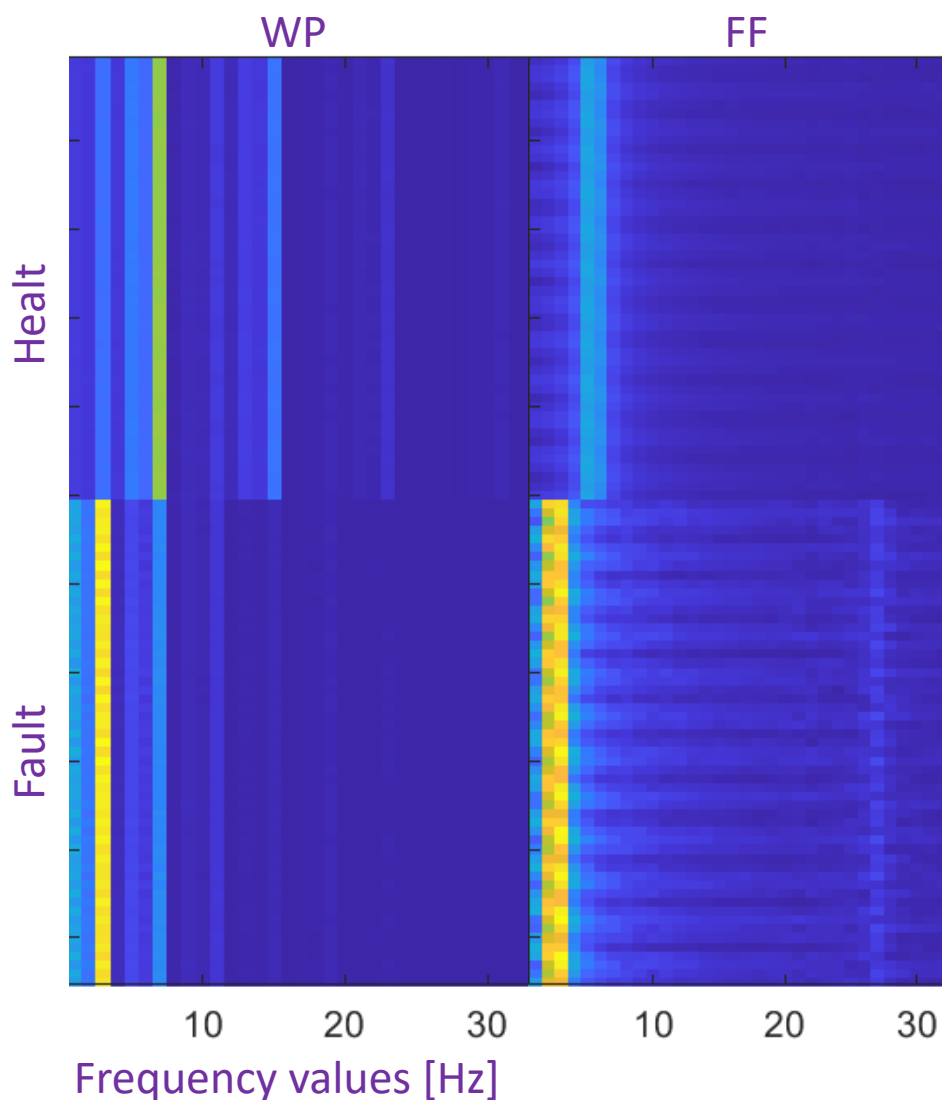


Figure 4-3 Heat map chart comparing healthy and bearing fault dataset at 700 RPM, load of 1.5kW, Depth 10 and window length of 12500

4.2.1 FFT Code

```
84     if strcmp(Aproach,'fft')==1
85         while (pos+WinSize<=Sig_Length)
86             y = data(pos:pos+WinSize-1).*w';
87             Res = Sample_rate/power(2,(10-Depth+1));
88             NFFT = 2^nextpow2(Res);
89             YB = fft(y,NFFT)/Res;
90             FFTB = 2*abs(YB(1:NFFT/2));
91             FH1(f,:) = FFTB;
92
93             TimeB(1,:) = mean(y);
94             TimeB(2,:) = var(y);
95             TimeB(3,:) = skewness(y);
96             TimeB(4,:) = kurtosis(y);
97             TimeB(5,:) = rms(y);
98             TimeB(6,:) = max(y)/rms(y);
99             time(f,:) = TimeB;
100
101             f = f + 1;
102             pos = pos + WinSize/2;
103         end
104         characteristics = [time, FH1];
```

Figure 4-4 FFT Code

In the case of "FFT" (Figure 4-4), I began by creating a while loop that iterates over the data in segments. The loop continues until the end of the signal length is reached, which is the length of the dataset for the given file. The window length increments at the end by half of itself.

Inside the loop, the segment of the signal (y) is extracted using the current position (pos) and the window size (WinSize). This

segment is multiplied by a window function w, which is used to reduce spectral leakage and improve the frequency resolution of the subsequent analysis.

Then, I create an NFFT variable that returns the exponents for the smallest powers of two that satisfy each element in the resolution. This pads the signal, which can speed up the computation of the FFT when the signal length is not an exact power of 2.

After that, I take a window of data (y) with NFFT and apply FFT to it using the MATLAB function "fft". By applying FFT to (y) with NFFT, we ensure that:

- The length of the segment y is extended to NFFT;
- The FFT result has the desired frequency resolution.

We take advantage of the efficiency of FFT computations with lengths that are powers of 2.

After applying the FFT, I normalize the result by dividing it by the resolution. Then, I calculate the one-sided amplitude spectrum FFTB by taking the absolute value of the FFT and multiplying it by 2, only considering the first half of the FFT result (up to NFFT/2). Next, I store the calculated FFT

characteristics in the variable FH1 for further processing. Additionally, I compute time-domain features for each segment of data (y). These features include:

- The mean of the window (TimeB(1,:));
- The variance of the window (TimeB(2,:));
- The skewness of the window (TimeB(3,:));
- The kurtosis of the window (TimeB(4,:));
- The root mean square (RMS) of the window (TimeB(5,:));
- The ratio of the maximum value to the RMS of the window (TimeB(6,:)).

These features are calculated using MATLAB functions such as mean, var, skewness, kurtosis, and rms.

Finally, I store the time-domain features in the variable time. The loop continues until all segments of the data have been processed. At the end of the loop, the characteristics variable is updated to include both the time-domain features (time) and the FFT characteristics (FH1). These characteristics are used for further analysis and classification.

4.2.2 WPT Code

```

105 elseif strcmp(Aproach,'wpt')==1
106     while (pos+WinSize<=Sig_Length)
107         y = data(pos:pos+WinSize-1).*w';
108
109         TH1a = wpdec(y,Depth,'db1');
110         [DP_PalH,DP_SeqH] = otnodes(TH1a,'dp');
111         EH1a = wenergy(TH1a);
112         EH1(f,:) = EH1a;
113
114         TimeH(1,:) = mean(y);
115         TimeH(2,:) = var(y);
116         TimeH(3,:) = skewness(y);
117         TimeH(4,:) = kurtosis(y);
118         TimeH(5,:) = rms(y);
119         TimeH(6,:) = max(y)/rms(y);
120         time(f,:) = TimeH;
121
122         f = f + 1;
123         pos = pos + WinSize/2;
124     end
125     characteristics = [time, EH1];
126 end

```

In the case of "WPT" (Figure 4-5), the implementation followed a similar process with a different signal processing technique. Similarly, I used a while loop to iterate over the data in segments, extracting the segment of the signal (y) and multiplying it by a window function (w) to enhance frequency resolution.

However, instead of applying FFT, I applied the Wavelet

Figure 4-5 WPT Code

Packet Transform (WPT) to the windowed data (y) using the MATLAB function “wpdec”. In this case “Haar” wavelet family is used (db1) but I have tested with mother wavelet as well (db10). This transformed the signal into wavelet packets, capturing both time and frequency information simultaneously.

From the wavelet packet decomposition (TH1a), I extracted node and sequence information using the “otnodes” function, which was crucial for analysing the structure of the signal. Then, I calculated the energy of the wavelet packet coefficients using the “wenenergy” function, representing the distribution of signal energy across different frequency components.

Additionally, I computed time-domain features for each window of data (y), including mean, variance, skewness, kurtosis, and rms, similar to the FFT approach.

At the end of the loop, both time-domain features (time) and wavelet packet energy features (EH1) were stored in the characteristics and sent to data [6] for further analysis and classification.

4.3 Machine learning code

For the machine learning code (Figure 4-6), I utilized the t_data variable within the file loop to record each file labeled as healthy at 700 RPM, ensuring it could be used for training the machine learning (ML) model.

```

36         if strcmp(data{i,1}, 'Healthy') && str2double(data{i,4}) ~= 2 && str2double(data{i,2}) == 700
37             t_data = [t_data; data{i,6}];
38         elseif ~strcmp(data{i,1}, 'Healthy') && str2double(data{i,2}) == 700
39             f_data = [f_data; data{i,6}];
40         elseif strcmp(data{i,1}, 'Healthy') && str2double(data{i,4}) == 2 && str2double(data{i,2}) == 700
41             h_data = [h_data; data{i,6}];
42
43         end
44     end
45 end
46 [T_Size, ~] = size(t_data);
47 [F_Size, ~] = size(f_data);
48 [H_Size, ~] = size(h_data);
49 TrainLabel = ones(T_Size, 1);
50 FaultyLabel = -ones(F_Size, 1);
51 HealthyLabel = ones(H_Size, 1);
52 HF = [h_data; f_data];
53 HFLabel = [HealthyLabel; FaultyLabel];
54 argument = ['-s 2 -t 2 -n 0.01 -g 0.1'];
55 model = svmtrain(TrainLabel, t_data, argument);
56 [predict_label, accuracy, dec_values] = svmpredict(HFLabel, HF, model);
57
58 TP = Count(predict_label(1:H_Size), '>0');
59 FP = H_Size - TP;
60 TN = Count(predict_label(H_Size+1:H_Size+F_Size), '<0');
61 FN = F_Size - TN;
62 BER = ((FP/(FP+TP)+FN/(FN+TN))/2);

```

Figure 4-6 ML Code

I selected only the first and last sets, leaving the second set to be recorded by h_data for healthy data testing later. Similarly, I created a variable f_data to collect all faults at 700 RPM for testing. After recording the data, I calculated the size of each data set to prepare for labeling. I assigned training labels to the training data, marking them all as healthy by assigning a value of 1. Likewise, I labeled the healthy data with 1 and the faulty data with -1. Then, I combined the healthy and faulty labels into HFLable, and the healthy and faulty data into HF, forming our testing dataset.

Next, the SVM model was trained using the “svmtrain” function with the provided training labels, training data, and SVM settings (argument). Subsequently, the trained model was employed to predict labels for the combined test dataset (HF) using the “svmpredict” function.

Finally, based on the predicted labels and the actual labels, the numbers of true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN) are calculated. To evaluate the accuracy of the machine learning model, we used these values to find the balanced error rate (BER).

5. TEST and Discussion

5.1 Test

5.1.1 FFT

I initially conducted tests using FFT due to its computational efficiency, on depth of 6 and window length of 5000. I focused on data at 700 RPM, using the first and second sets of healthy data for training, and the third set for testing (Figure 5-1). This resulted in a BER of 11%, which is acceptable. However, the ML model tended to misclassify healthy signals as faults, and it incorrectly labelled the bearing fault at 0.75 load as healthy, with some mislabelling observed particularly for eccentricity.

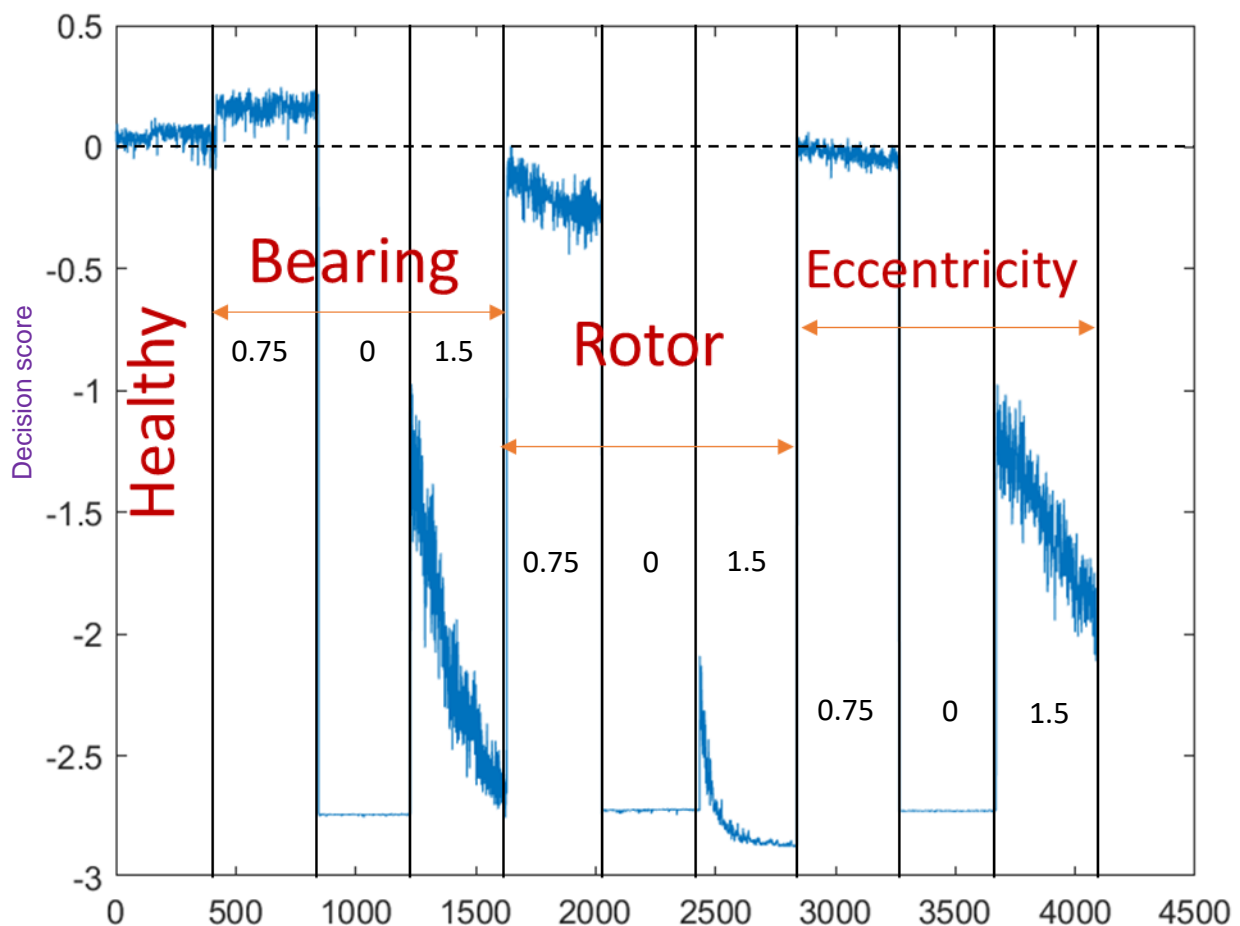


Figure 5-1 Decision Score (test 1 healthy 1, 2, FFT)

I decided to adjust the training data by using the first and third sets of healthy data for training and the second one for testing, while keeping the depth at 6, window length at 5000, and RPM at 700, again using FFT. This resulted in a higher accuracy with a BER of 8%, with fewer healthy data being flagged as faulty (Figure 5-2). However, the bearing fault at 0.75 load is still flagged as healthy, and there are still some signals mislabelled for the eccentricity at load 0.75.

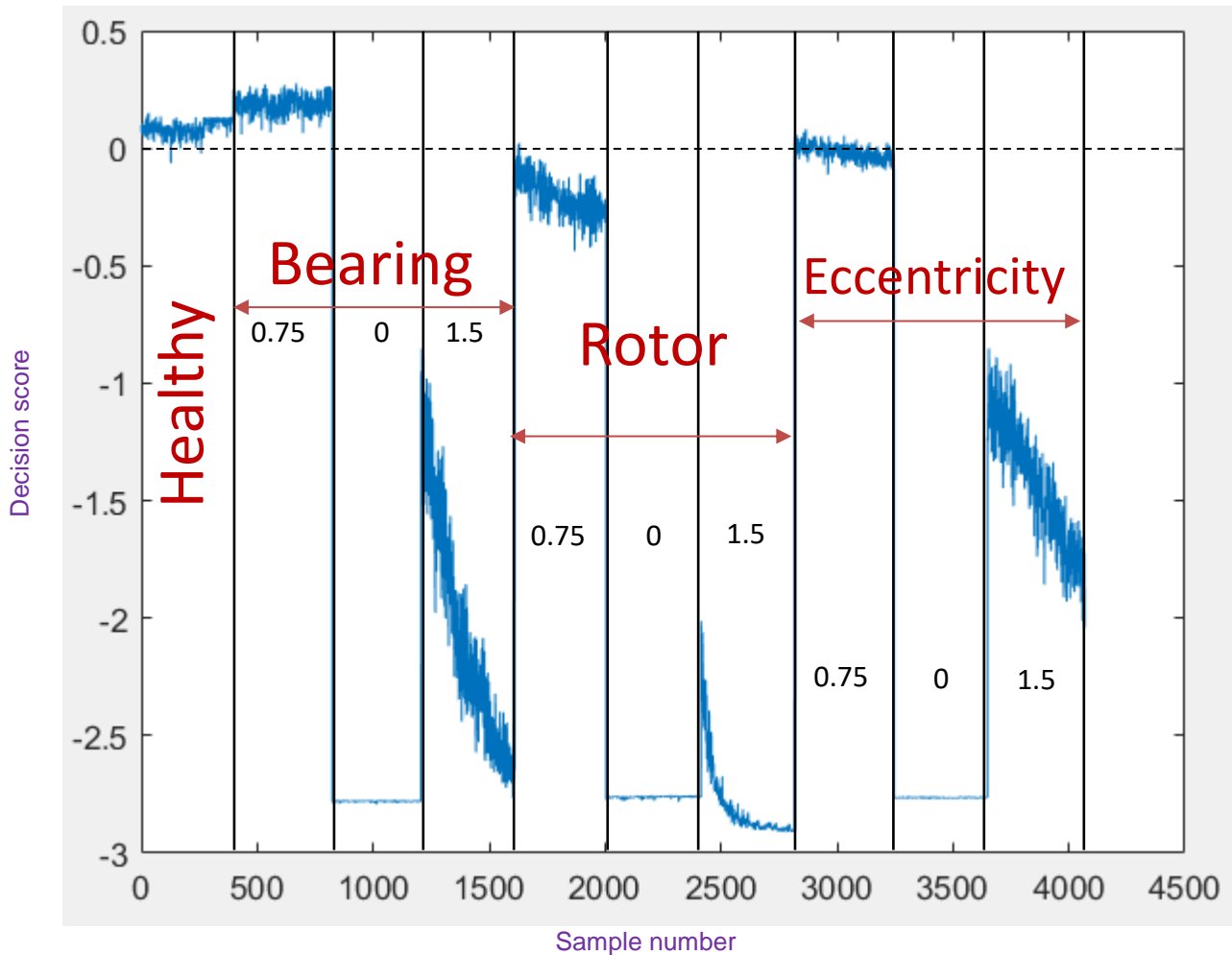


Figure 5-2 Decision Score (test 2 healthy 1, 3, FFT)

I continued testing by using only the second and third healthy data as the training set, but I didn't succeed in increasing the accuracy and received 12% BER. For following tests, I will use the first and third healthy data set for training.

To try to reduce the BER, I began by adjusting the depth to achieve different resolutions. After computing depths from 1 to 11, I found that beyond a depth of 10, the quality of classification decreased. Therefore, I decided not to test further depths. Next, I experimented with changing the window length, ranging from 1250 to 20000 with sensible intervals, to see if it would improve the

results. I found the most accuracy at window length of 12500 and depth of 10, which will be used for following test. Both increasing and decreasing the window length or depth from that point led to higher balanced error rates. The most accurate window length was 12500 and most of the widows were best performing at depth of 9 (Figure 5-3).

During testing, I observed that when the window length increased the higher depth was required to receive different results. For example, changing the depth from 1 to 8 at WL of 10000 didn't make a difference. However, increasing the depth above 4 at 1250 window length made a difference.

Depth \ Window L	1	2	3	4	5	6	7	8	9	10	11
20000	7.50%	7.50%	7.50%	7.50%	7.50%	7.50%	7.50%	7.50%	7.28%	7.33%	8.09%
15000	6.98%	6.98%	6.98%	6.98%	6.98%	6.98%	6.98%	6.98%	6.98%	7.02%	8.73%
12500	6.79%	6.79%	6.79%	6.79%	6.79%	6.79%	6.79%	6.79%	6.72%	6.62%	9.30%
10000	7.77%	7.77%	7.77%	7.77%	7.77%	7.77%	7.77%	7.77%	7.66%	8.89%	9.44%
5000	8.09%	8.09%	8.09%	8.09%	8.09%	8.09%	8.09%	8.57%	9.95%	12.80%	10.09%
2500	11.36%	11.36%	11.36%	11.36%	11.36%	11.28%	10.61%	12.44%	14.53%	10.64%	10.09%
1250	15.29%	15.29%	15.29%	15.29%	15.28%	15.28%	15.66%	15.39%	14.81%	14.81%	14.97%
BER%	Settings: FFT 700 RPM All Loads							Best BER% at X axis		Best BER% at Y axis	

Figure 5-3 Tests on different Depth and Window length

Once I identified the optimal training set and the most accurate combination of load and depth, I decided to experiment with a different combination of RPMs and loads (Figure 5-4). I started my test by using a particular speed and one load. Then I expanded to two different loads and one speed, and vice versa, until I was training and testing with the entire dataset.

RPM \ Load	0.75	0	1.5	0.75/0	0.75/1.5	0/1.5	0.75/0/1.5
700	19.50%	2.00%	0.00%	9.95%	10.11%	0.99%	6.62%
1000	23.68%	45.59%	30.00%	28.07%	26.59%	38.12%	28.79%
1500	47.49%	47.19%	34.14%	45.61%	37.46%	36.61%	41.00%
700/1000	26.99%	43.98%	14.89%	32.44%	26.25%	29.57%	28.91%
700/1500	35.03%	42.68%	18.11%	37.18%	26.37%	33.54%	33.07%
1000/1500	35.35%	46.76%	31.23%	40.19%	38.68%	37.14%	40.74%
1500/1000/700	31.41%	47.19%	21.67%	39.52%	33.82%	37.37%	38.63%
BER%			Best BER% at X axis		Best BER% at Y axis		

Figure 5-4 Tests on different RPMs and Loads

The machine learning algorithm achieved its highest accuracy at 700 rotations per minute. The easiest load to detect was mostly at 1.5 kW. Notably, it faced the most difficulty detecting faults at the highest speed and medium load (1500 RPM, 0.75 kW). Training on the entire dataset yielded decent accuracy. Further tests revealed that increasing the frequency resolution for larger datasets improves accuracy. Increasing the depth to 15 reduced the bit error rate (BER) to 27% for the entire dataset, compared to the original 38.63%. I intended to try a depth of 20 but quickly realized my computer might not handle it. The whole operating system slowed down, and I couldn't even move my mouse.

Another key point in that table is the conducted experiment with load of 1.5 kW at 700 RPM, which

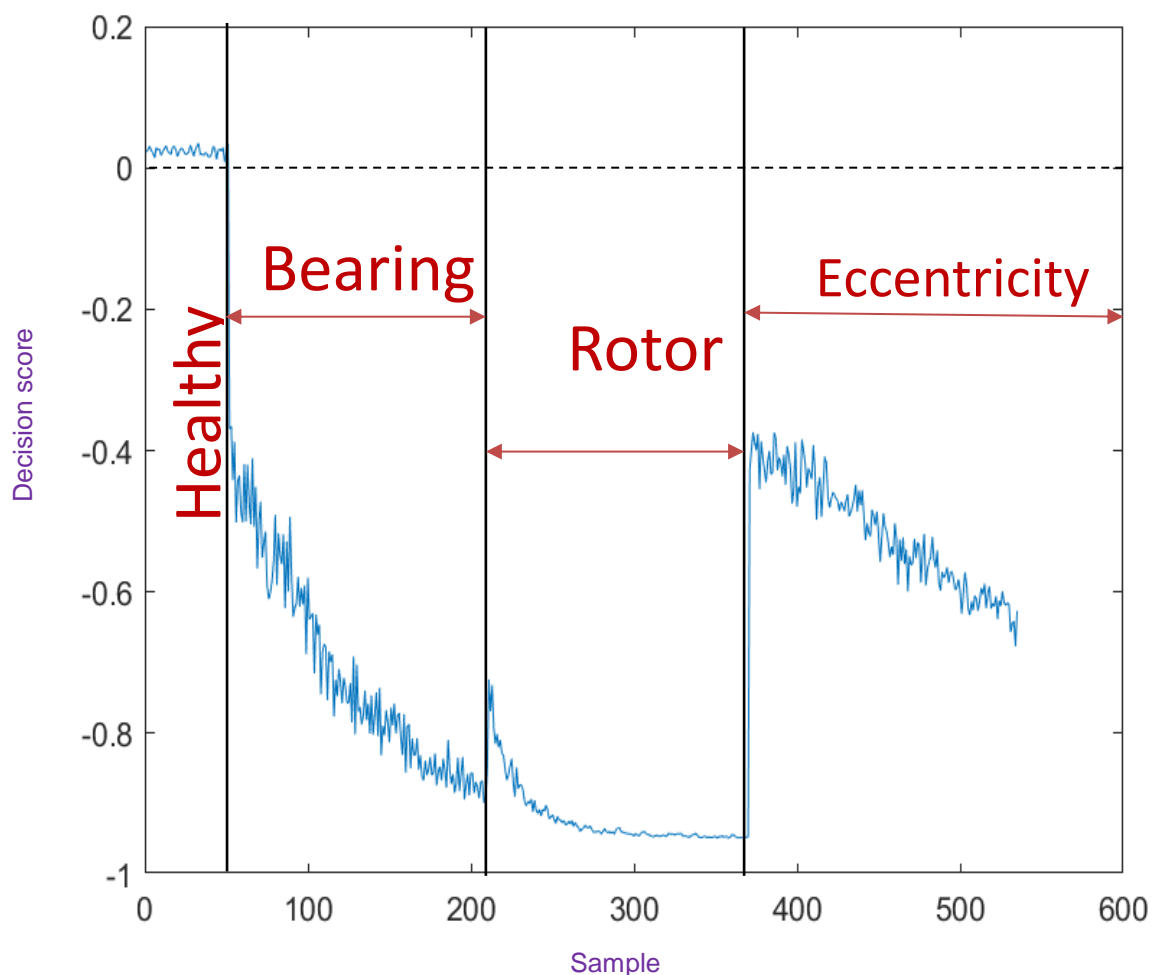


Figure 5-5 Perfect decision Score at 700 RPM using Load 1.5 and training on healthy dataset 1 and 3. FFT approach.

resulted in ideal perfect accuracy (Figure 5-5) with a 0% BER. This means the model performed exceptionally well in classifying faults under these conditions.

5.1.2 WPT

In WPT computing, speed posed a significant challenge. Processing the easiest dataset at the lowest depth took around a minute and a half, and at higher depths, this could extend beyond 10 minutes. Due to the time-consuming nature of the process and the project's time constraints, I computed the best results I obtained from FFT using WPT. I started by attempting to replicate the perfect accuracy using a speed of 700 RPM and a load of 1.5 kW, and successfully achieved the same BER of 0% (Figure 5-6). Compared to FFT there is more attenuation in the decision score.

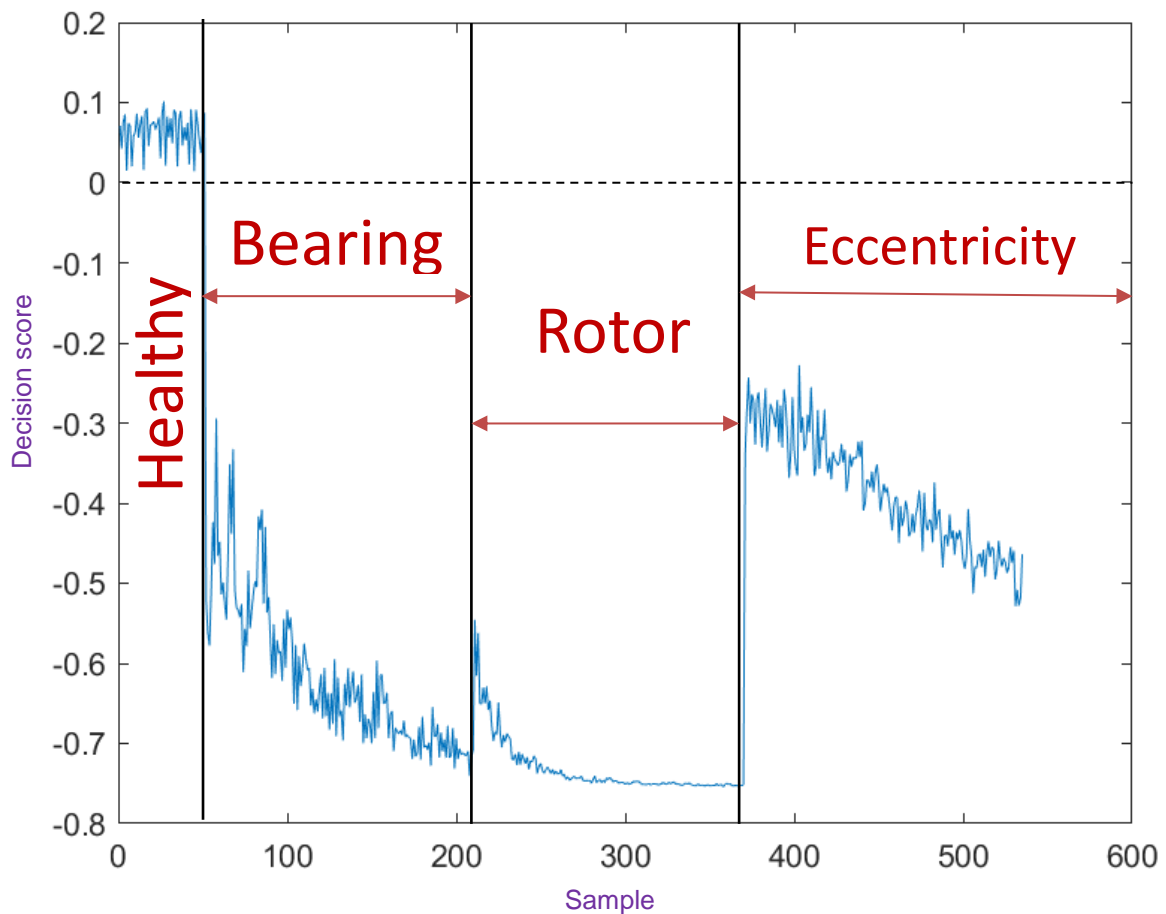


Figure 5-6 Perfect decision Score at 700 RPM using Load 1.5 and training on healthy dataset 1 and 3. WPT approach.

Later, I conducted additional tests at a depth of 6 and a window length of 5000 for computational efficiency and achieved significant accuracy with WPT, which I compared to FFT (Figure 5-7).

Settings \ Approach	WPT	FFT
700 RPM 0 kW	0.00%	1.17%
700 RPM 0 & 1.5 kW	0.00%	0.58%
700 RPM All Loads	8.44%	8.09%
1000 RPM 0 kW	33.08%	47.40%
1500 RPM 1.5 kW	49.29%	32.75%
	BER%	

Figure 5-7 WPT and FFT comparison

These tests also revealed that with smaller datasets, the classifier's accuracy improves with lower depth and window length. For example, FFT at 700 RPM and 0 kW load showed better accuracy compared to what we observed in Figure 5-4, but errors increased with larger datasets, such as the 700 RPM all loads dataset, which resulted in lower accuracy.

Another interesting result is that for the whole dataset, WPT achieved an accuracy of 27% BER at a depth of 10, which is significantly more accurate than FFT with the same settings. It matched the accuracy I achieved for FFT with a depth of 15. WPT performed better overall but was more computationally challenging.

5.2 Discussion

The system at 700 RPM had the most trouble finding the bearing fault and some problems with finding the eccentricity fault both at 0.75 load but had no trouble with the other faults and loads.

Among all RPMs, 700 RPM was the most accurate, and among all loads, 1.5 kW was the most accurate.

When using both specific RPM and Load the system is more accurate even achieving perfect accuracy at 700 RPM and 1.5 load.

With larger datasets, greater depth and window length prove to be more accurate, while with smaller datasets, lower depth and window length prove to be more accurate.

6. PROJECT EVALUATION/CONCLUSION

6.1 Evaluation

The project aimed to develop a fault detection system for motor systems using signal processing techniques and machine learning algorithms. The system's primary objectives were to accurately identify faults in motor systems and distinguish them from healthy operation. Different tests were done using two signal processing approaches and many different combinations of RPMs, loads and training datasets. Finding an ideal BER at each stage. The best results were achieved at a speed of 700 RPM and a load of 1.5 kW. If this system were to be implemented now, a depth of 10 and a window length of 11250 would be used. I successfully developed a fault detection system for motor systems, leveraging signal processing techniques and machine learning algorithms. The system demonstrated high accuracy in fault detection, with optimal parameter settings identified for different techniques. Further optimization and real-world testing are needed to validate the system's performance in industrial applications.

6.2 Future Work

In the future, there are several areas I plan to explore to enhance the fault detection system for motor systems. Firstly, I aim to conduct further testing with the Wavelet Packet Transform (WPT) approach. While initial testing showed promise, additional experiments are needed to optimize the WPT technique and understand its full potential. It's worth noting that WPT requires a significant amount of computing power and time to process, which limited the scope of experiments in this study.

Additionally, I intend to explore alternative signal processing approaches beyond WPT. Techniques such as Empirical Mode Decomposition (EMD) or Time-Frequency Analysis could provide valuable insights into motor system behaviour and improve fault detection accuracy.

A crucial aspect for improving accuracy is training the machine learning models on a larger scale of healthy dataset. By expanding the healthy dataset, the models can learn more representative patterns, enhancing their ability to differentiate between healthy and faulty states.

Feature scaling is another area I plan to focus on. Implementing feature scaling techniques can normalize or standardize features, leading to better model performance. Additionally, utilizing

feature scaling for faulty features may help in acquiring more accurate labelling, improving overall system accuracy.

Furthermore, I aim to integrate real-time feature extraction and fault detection capabilities into the system. This will enable proactive fault detection and preventive maintenance, enhancing the overall reliability of motor systems.

6.3 Conclusion

In conclusion, the development of a fault detection system for motor systems has shown promising results in identifying and classifying faults. Through the utilization of signal processing techniques such as the Fast Fourier Transform (FFT), Wavelet Packet Transform (WPT) and machine learning algorithms like Support Vector Machines (SVM), significant progress has been achieved in automating fault detection processes.

Experiments revealed the effectiveness of the FFT approach in capturing frequency features and the SVM algorithm in accurately classifying faults. By fine-tuning parameters such as depth, training dataset and window length, it was possible to optimize the performance of the system and achieve perfect accuracy (BER of 0%).

WPT offers greater accuracy but requires more computing time. It has demonstrated better performance with larger scale data compared to FFT. Further tests are needed to determine performance on different data.

7. REFERENCES

- [1] Hughes, A., & Drury, B. (2013). *Electric motors and drives: Fundamentals, types and applications*. Elsevier Science & Technology. [1]
- [2] E. Smart, D. Brown and L. Axel-Berg, "Comparing one and two class classification methods for multiple fault detection on an induction motor," *2013 IEEE Symposium on Industrial Electronics & Applications*, Kuching, Malaysia, 2013, pp. 132-137, doi: 10.1109/ISIEA.2013.6738982.
- [3] M. A. Sheikh, N. R. Saad, N. b. Mohd Nor, S. Tahir Rakhsh and M. Irfan, "An Unsupervised Automated Method to Diagnose Industrial Motors Faults," *2018 IEEE International Conference on Electrical Systems for Aircraft, Railway, Ship Propulsion and Road Vehicles & International Transportation Electrification Conference (ESARS-ITEC)*, Nottingham, UK, 2018, pp. 1-7, doi: 10.1109/ESARS-ITEC.2018.8607646.
- [4] Sheikh, M.A., Bakhsh, S.T., Irfan, M. *et al.* A Review to Diagnose Faults Related to Three-Phase Industrial Induction Motors. *J Fail. Anal. and Preven.* **22**, 1546–1557 (2022). <https://doi.org/10.1007/s11668-022-01445-2>
- [5] B. Li, M. Chow, Y. Tipsuwan, and J. Hung, "Neural-network-based motor rolling bearing fault diagnosis," *Industrial Electronics, IEEE Transactions on*, vol. 47, no. 5, pp. 1060–1069, 2000
- [6] Purushottam Gangsar, 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*, Volume 144, 2020, 106908, ISSN 0888-3270, <https://doi.org/10.1016/j.ymssp.2020.106908>. (<https://www.sciencedirect.com/science/article/pii/S0888327020302946>)
- [7] Strömbergsson D, Marklund P, Berglund K, Larsson P-E. Bearing monitoring in the wind turbine drivetrain - A comparative study of the FFT and wavelet transforms. *Wind Energy*. 2020; 23: 1381–1393. <https://doi.org/10.1002/we.2491>
- [8] Wakefield, Katrina. "A guide to the types of machine learning algorithms and their applications." SAS Inst. UK (2021). https://www.sas.com/en_gb/insights/articles/analytics/machine-learning-algorithms.html
- [9] Bouzida A, Touhami O, Ibtouen R, Belouchrani A, Fadel M, Rezzoug A (2010) Fault diagnosis in industrial induction machines through discrete wavelet transform. *IEEE Trans Industr Electron* 58(9):4385–4395
- [10] Leal MM, Costa FB, Campos JTLS (2019) Improved traditional directional protection by using the stationary wavelet transform. *Int J Electric Power Energy Syst* 105:59–69
- [11] Jaros, R., Byrtus, R., Dohnal, J. *et al.* Advanced Signal Processing Methods for Condition Monitoring. *Arch Computat Methods Eng* 30, 1553–1577 (2023). <https://doi.org/10.1007/s11831-022-09834-4>

- [12] M. Alonso-González, V. G. Díaz, B. L. Pérez, B. C. P. G-Bustelo and J. P. Anzola, "Bearing Fault Diagnosis With Envelope Analysis and Machine Learning Approaches Using CWRU Dataset," in IEEE Access, vol. 11, pp. 57796-57805, 2023, doi: 10.1109/ACCESS.2023.3283466.
- [13] Mohammad F. Yakhni, Sebastien Cauet, Anas Sakout, Hassan Assoum, Erik Etien, Laurent Rambault, Mohamed El-Gohary, Variable speed induction motors' fault detection based on transient motor current signatures analysis: A review, Mechanical Systems and Signal Processing, Volume 184, 2023, 109737, ISSN 0888-3270, <https://doi.org/10.1016/j.ymssp.2022.109737>.
- [14] Chih-Chung Chang and Chih-Jen Lin, LIBSVM : a library for support vector machines. ACM Transactions on Intelligent Systems and Technology, 2:27:1--27:27, 2011. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>
- [15] S. E. Pandarakone, S. Gunasekaran, Y. Mizuno and H. Nakamura, "Application of Naive Bayes Classifier Theorem in Detecting Induction Motor Bearing Failure," 2018 XIII International Conference on Electrical Machines (ICEM), Alexandroupoli, Greece, 2018, pp. 1761-1767, doi: 10.1109/ICELMACH.2018.8506836.
- [16] GeeksforGeeks. (2024). Understanding One-Class Support Vector Machines. [online] Available at: <https://www.geeksforgeeks.org/understanding-one-class-support-vector-machines/> [Accessed 18 Mar. 2024].
- [17] D. -J. Choi, J. -H. Han, S. -U. Park and S. -K. Hong, "Comparison of motor fault diagnosis performance using RNN and K-means for data with disturbance," 2020 20th International Conference on Control, Automation and Systems (ICCAS), Busan, Korea (South), 2020, pp. 443-446, doi: 10.23919/ICCAS50221.2020.9268271.
- [18] Kowalski, Czeslaw T., and Teresa Orłowska-Kowalska. "Neural networks application for induction motor faults diagnosis." Mathematics and computers in simulation 63.3-5 (2003): 435-448.
- [19] The MathWorks, Inc. (2022). MATLAB version: 23.2.0.2409890 (R2023b). Available: <https://www.mathworks.com>.
- [20] Bishop, Owen. Electronics - Circuits and Systems : Circuits and Systems, Taylor & Francis Group, 2011. ProQuest Ebook Central, <https://ebookcentral.proquest.com/lib/portsmouth-ebooks/detail.action?docID=645028>.
- [21] traction.com. (n.d.). Induction Motors: A Revolution in the Industry - TRACTIAN. [online] Available at: <https://traction.com/en/blog/induction-motors-revolutionizing-the-industry>.
- [22] Daware, K. (n.d.). Standard types of squirrel cage motors. [online] Available at: <https://www.electricaleasy.com/2014/02/standard-types-of-squirrel-cage-motors.html> [Accessed 2 Mar. 2021].

LIST OF ACHIEVEMENTS

I have successfully developed code that takes raw signal data, consisting of currents, applies windowing, and then utilizes two different types of signal processing techniques: FFT and WPT. Furthermore, I have successfully trained a single-class machine learning classifier to process and label the data. Through various tests, I achieved perfect accuracy on one of the tests.

Project Achievements

1. Successfully taken raw current data signals and signal processing applied
2. Successfully trained and tested a one-class SVM
3. Perfect accuracy achieved at given settings

Learning Achievements

1. I learned to program in MATLAB.
2. I learned how to apply windowing.
3. I enhanced my understanding for signal processing approaches and how to apply them.
4. Deeper understanding of machine learning approaches and types of ML's.
5. I learned the importance of Induction motors in today's world.