

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO CEARÁ
PRÓ-REITORIA DE PESQUISA, PÓS-GRADUAÇÃO E INOVAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Rebeca Guerreiro Carvalho Cunha

**Machine Learning and Multiresolution Signal Decomposition for
Embedded Applications: A Study of Short-Circuit Failure Detection
in Induction Motors**

Fortaleza – CE

October 20, 2019

Rebeca Guerreiro Carvalho Cunha

Machine Learning and Multiresolution Signal Decomposition for Embedded Applications: A Study of Short-Circuit Failure Detection in Induction Motors

Dissertação submetida ao Programa de Pós-Graduação em Ciência da Computação do Instituto Federal de Educação, Ciência e Tecnologia do Ceará, como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

Linha de Pesquisa: Engenharia de Software

Supervisor: Elias Teodoro da Silva Júnior

Co-supervisor: Cláudio Marques de Sá Medeiros

Instituto Federal de Educação, Ciência e Tecnologia do Ceará (IFCE)

Pró-Reitoria de Pesquisa, Pós-Graduação e Inovação (PRPI)

Programa de Pós-Graduação em Ciência da Computação (PPGCC)

Fortaleza – CE

October 20, 2019

Dados Internacionais de Catalogação na Publicação
Instituto Federal do Ceará - IFCE
Sistema de Bibliotecas - SIBI
Ficha catalográfica elaborada pelo SIBI/IFCE, com os dados fornecidos pelo(a) autor(a)

G934m Guerreiro Carvalho Cunha, Rebeca.

Machine Learning and Multiresolution Signal Decomposition for Embedded Applications : A Study of Short-Circuit Failure Detection in Induction Motors / Rebeca Guerreiro Carvalho Cunha. - 2019.
145 f. : il. color.

Dissertação (Mestrado) - Instituto Federal do Ceará, Mestrado em Ciência da Computação, Campus Fortaleza, 2019.

Orientação: Prof. Dr. Elias Teodoro da Silva Junior.

Coorientação: Prof. Dr. Cláudio Marques de Sá Medeiros.

1. Apendizado de Máquinas. 2. Detecção de Falhas. 3. Motor de Indução. 4. Sistemas Embarcados. 5. Transformada Wavelet Discreta. I. Título.

CDD 004

REBECA GUERREIRO CARVALHO CUNHA

Aprendizado de máquinas e decomposição multiresolução de sinal para aplicações
embarcadas

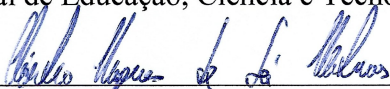
Dissertação submetida à Coordenação do Curso de Pós-graduação em Ciência da Computação do Instituto Federal de Educação, Ciência e Tecnologia do Ceará, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação, área de concentração Ciência da Computação.

Aprovada em 20 / 08 / 2019.

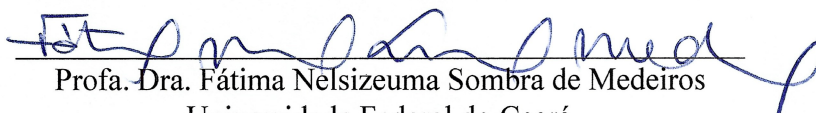
BANCA EXAMINADORA



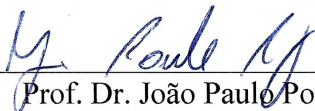
Prof. Dr. Elias Teodoro da Silva Júnior (Orientador)
Instituto Federal de Educação, Ciência e Tecnologia do Ceará - IFCE



Prof. Dr. Cláudio Marques de Sá Medeiros (Co-orientador)
Instituto Federal de Educação, Ciência e Tecnologia do Ceará - PPGCC/IFCE



Profa. Dra. Fátima Nelsizeuma Sombra de Medeiros
Universidade Federal do Ceará



Prof. Dr. João Paulo Pordeus Gomes
Universidade Federal do Ceará

Acknowledgements

First, I would like to thank God for providing me the guidance, strength and capability to proceed successfully.

I will always be grateful to my mother Carmem, which encouraged me since I was a child to pursue my dreams and objectives. And for convincing me I could do anything if I put enough effort to it. You will always be my example of determination and kindness.

I would like to thank my husband Vinicius for being supportive and patient.

I'm very grateful to my supervisors Prof. Elias Teodoro and Prof. Claudio Sá for all the guidance, support and teaching since undergraduate studies, you are the best. I will take your examples of commitment, respect and professionalism to my life.

I'm also grateful to the Federal Institute of Ceará, specially the professors and colleagues of PPGCC, LIT and LPE, for all the support and teachings.

“In theory, there is no difference between theory and practice. But, in practice, there is.”
(Benjamin Brewster)

Abstract

With the advances in embedded systems and machine learning, these fields of computer science become present in most people daily lives, often unnoticed. In the last years, those two areas could be combined for the development of very diverse applications, including health care, farming and manufacturing process. In many cases, the study of the development of classification and regression models that might be executed by embedded systems is done without taking into consideration some aspects related to data acquisition and processing, which are fundamental to the project viability or have a significant impact in important requirements like execution time, power consumption and cost. Thus, this text describes a methodology for the development of embedded applications combining signal processing and machine learning, oriented to inter-turn short-circuit failure detection in three-phase induction motors. A test-bench, with a modified motor which short-circuits are emulated, is used and the voltage induced by motor's axial flux signal is acquired in different operating conditions and failure levels, to compose the dataset used for training and testing the model. In this stage, Higher order statistics and dispersion measurements calculated from the time domain signal are used to compose the features set. Multiresolution analysis, using the discrete wavelet transform, is applied to improve the features extraction routine. With this technique, the signal is decomposed in frequency sub-bands. Different classifiers are used, among which SVM, OPF, MLP and KNN stand out. Analysis of Variance (ANOVA) is used to select features, seeking to improve the failure detection and reduce the classification task computational cost. 99.23 % accuracy for 7 classes (healthy and 6 failure sub-types) and 99.93 % for 2 classes (healthy x defective) are achieved. Modifications in the dataset are done to reduce the computational cost of the classification task. The chosen model is embedded on a microcontroller with an ARM processor and it is performed an analysis of the embedded implementation using execution time and memory footprint.

Key-words: Discrete Wavelet Transform. Embedded Systems. Failure Detection. Induction Motor. Machine Learning.

Resumo

Com os avanços nas áreas de sistemas embarcados e aprendizado de máquinas, esses dois campos da computação tornaram-se presentes no cotidiano da maioria das pessoas, muitas vezes não sendo nem percebidos. Nos últimos anos, essas duas áreas puderam ser combinadas para o desenvolvimento de aplicações bem diversas que vão desde soluções médicas e de cuidados com a saúde, à agropecuária e industriais. Em grande parte dos casos, o estudo para o desenvolvimento de modelos de classificação ou regressão de aplicações que deverão ser executadas por sistemas embarcados acontece sem levar em considerações aspectos relacionados a aquisição e processamento de dados que são fundamentais para viabilidade dos projetos ou que tem um forte impacto em requisitos importantes como tempo de execução, consumo energético e custo. Com isso em vista, é descrita neste trabalho, uma metodologia para o desenvolvimento de aplicações embarcadas que combinam processamento de sinal e aprendizado de máquinas, voltado ao problema de detecção de falhas de curto-circuito estatístico em motores de indução trifásicos. Sinais de tensão induzidas devido ao fluxo axial no motor são captados em diferentes condições de operação e níveis de falha, para a composição do conjunto de dados utilizado para o treinamento e teste do modelo. Para isso, uma bancada de testes que contém um motor modificado para permitir a emulação de curto-circuitos é utilizada. São avaliadas a utilização de estatística de alta ordem e medidas de dispersão para compor o conjunto de atributos, extraídos diretamente do sinal de fluxo no domínio do tempo, assim como a utilização da transformada wavelet discreta para realizar uma análise multiresolução, em que o sinal é decomposto em sub-bandas de frequência. Diferentes classificadores são avaliados, em que SVM, OPF, MLP e KNN se destacam. Análise de variância (ANOVA) é utilizada para seleção de atributos com o intuito de melhorar a detecção das falhas e diminuir o custo computacional da etapa de classificação. Foi alcançada acurácia para 7 classes (normal e 6 subtipos de falhas) de 99,23 % e para 2 classes (normal x falha) de 99,93 %. Adaptações nos conjuntos de dados são feitas para reduzir o custo computacional e otimizar a aplicação embarcada. O modelo escolhido foi embarcado em um microcontrolador com processador ARM e a análise da implementação embarcada é feita utilizando as métricas tempo de execução e utilização de memória RAM e FLASH.

Palavras-chave: Aprendizado de Máquinas. Detecção de Falhas. Motor de Indução. Sistemas Embarcados. Transformada Wavelet Discreta.

List of Figures

Figure 1 – Three-phase squirrel-cage motor	38
Figure 2 – Wavelet decomposition computed using the fast wavelet transform, with cascade of filterings $\bar{h}$ and $\bar{g}$	45
Figure 3 – McCulloch and Pits neuron model representation	48
Figure 4 – Multilayer perceptron with two hidden layers	51
Figure 5 – Neuron representation	52
Figure 6 – RBF neural network diagram	55
Figure 7 – Illustration of an optimal hyperplane for linearly separable patterns. The blue data points are the support vectors.	59
Figure 8 – Illustration of a soft margin hyperplane. One circle data point falls on the wrong side of the decision surface	60
Figure 9 – Mappings in a support vector machine for pattern classification	61
Figure 10 – OPF fit illustration	64
Figure 11 – OPF prediction illustration	64
Figure 12 – Machine Learning Embedded Application Design	71
Figure 13 – Test bench diagram	74
Figure 14 – Test bench photography	74
Figure 15 – Representation of emulated fault types	75
Figure 16 – Equipment used to emulate mechanical load	76
Figure 17 – Sensor placement diagrams	77
Figure 18 – Distribution of the patterns in the dataset	77
Figure 19 – Signal conditioning before analog to digital conversion	78
Figure 20 – Frequency spectrum of flux signal before and after analog low pass filtering	79
Figure 21 – Comparative of the axial flux for normal condition versus each type of failure. The motor operates without load, 60Hz	81
Figure 22 – Comparative of the axial flux for normal condition versus each type of failure. The motor operates with full load, 60 Hz	82
Figure 23 – Pair plots of the four features: Skewness, kurtosis, mean absolute devia- tion and median absolute deviation	83
Figure 24 – Simple perceptron accuracy boxplot	84
Figure 25 – Features histogram per class	85
Figure 26 – Bayesian classifiers accuracy box-plots for parameterization and test data sets, with different co-variance matrices	85
Figure 27 – Comparison of average accuracy among the evaluated classifiers	87
Figure 28 – Accuracy comparison of different features set and classifiers	89
Figure 29 – Flux approximation and detail coefficients for 2 level DWT using Db-2	97

Figure 30 – Comparison of different features set analyzed with KNN for 2 levels of decomposition	99
Figure 31 – Comparison of different features set (check Table 25 for the description), according to the number of DWT levels, with KNN	103
Figure 32 – Comparison of different features set (check Table 26 for the description), for 3 levels of decomposition with a KNN classifier	104
Figure 33 – Comparison of different features set (check Table 27 for the description), for 5 levels of decomposition with a KNN classifier	106
Figure 34 – Short-circuit failure detection proposed approach	111
Figure 35 – Evolution of the classification performance for different values of total acquisition time (TAT)	116
Figure 36 – Normal/failure accuracy and normal TP rate box-plot comparison for FS13 and FS20 over total acquisition time reduction	117
Figure 37 – Accuracy comparison for FS13, with and without the compensation in the number of levels depending on f_s	119
Figure 38 – Average results for different values of f_s	120
Figure 39 – Normal/failure accuracy and normal TP rate box-plot for FS20 over f_s reduction	120
Figure 40 – Results for simultaneous reduction in TAT and f_s using FS20 and SVM	122
Figure 41 – Total execution time distribution with FPU enabled	127
Figure 42 – RAM memory usage	127
Figure 43 – FLASH memory usage	128

List of Tables

Table 1 – Test set confusion matrix for SVM and Original FS (MeanAD, MedianAD, Skewness and Kurtosis)	88
Table 2 – Test Set confusion matrix for SVM using MeanAD, Skewness, Kurtosis as features	88
Table 3 – Average global accuracy comparison between Original FS and after medianAD is removed	88
Table 4 – Comparison between mean global accuracies with and without the load	90
Table 5 – Test set confusion matrix for SVM and Original FS + frequency + load	90
Table 6 – Test set confusion Matrix for SVM and Original FS + frequency	91
Table 7 – Test set confusion Matrix for OPF and Original FS + frequency + load	91
Table 8 – Test set confusion Matrix for OPF and Original FS + frequency	91
Table 9 – Test set confusion Matrix for MLP and Original FS + frequency + load	91
Table 10 – Test set confusion Matrix for MLP and Original FS + frequency	92
Table 11 – Teste set confusion matrix for KNN and Original FS + frequency + load	92
Table 12 – Test set confusion matrix for KNN and Original FS + frequency	92
Table 13 – SVM test set confusion matrix grouped by failure type	93
Table 14 – SVM test set confusion matrix grouped by amount of short turns	93
Table 15 – SVM test set confusion matrix with all the failures grouped	93
Table 16 – Frequency bands for two levels wavelet decomposition	96
Table 17 – Features set description	98
Table 18 – Test set confusion matrix for FS4 + KNN	99
Table 19 – Classification results for FS4	100
Table 20 – Test set binary confusion matrix for FS4	100
Table 21 – Test set confusion matrix for FS4 + SVM	100
Table 22 – Test set confusion matrix for FS4 + OPF	100
Table 23 – Test set confusion matrix for FS4 + MLP	101
Table 24 – Frequency bands for the wavelet coefficients	101
Table 25 – Features set description - increase in the decomposition levels	102
Table 26 – Features set description - selection using ANOVA for 3 levels of decomposition	104
Table 27 – Features set description - selection using ANOVA for 5 levels of decomposition	105
Table 28 – Classification results for FS13 and FS20	106
Table 29 – Test set binary confusion matrices for all best combinations of features set and classifiers	107
Table 30 – Test set confusion matrix for the combination FS20 + SVM	108

Table 31 – Test set confusion matrix for the combination FS13 + SVM	108
Table 32 – Test set confusion matrix for the combination FS20 + OPF	108
Table 33 – Test set confusion matrix for the combination FS20 + KNN	109
Table 34 – Test set confusion matrix for the combination FS20 + MLP	109
Table 35 – Summary of features and classifiers results considering 7 classes	110
Table 36 – Summary of preliminary study considering only normal and failure labels for classification	114
Table 37 – Test set average confusion matrix for FS20+SVM	114
Table 38 – Comparison of support vectors number	115
Table 39 – Number of points for the last level of decomposition for each combination of TAT and FS	117
Table 40 – Decomposition levels adjusted to f_s reduction for FS13	118
Table 41 – Average confusion matrix for Feature set 20 +SVM, with $f_s = 5$ kHz, $TAT = 3$ s $C = 10000$ and $\sigma = 0.25$	123
Table 42 – Proposed solution Parameters	123
Table 43 – Execution time comparison with and without the FPU	126
Table 44 – FLASH memory data distribution	128
Table 45 – Summary of the results. Training set = 80 % of the dataset, Dataset 3, K=5, FPU enabled	147

List of Algorithms

1	OVO (one-vs-one) SVM prediction	125
---	---	-----

List of abbreviations and acronyms

Acc	Accuracy
AD	Absolute Deviation
ADC	Analog to Digital Converter
AI	Artificial Intelligence
ANOVA	Analysis of Variance
A_x	Approximation Coefficients of Level x
ART	Adaptive Real-Time
ASIC	Application Specific Integrated Circuit
CBM	Condition Based Maintenance
CFS	Correlation Feature Selection
CMSIS	Cortex Microcontroller Software Interface Standard
CPU	Central Processing Unit
D_x	Detail Coefficients of Level x
DSP	Digital Signal Processing/Processor
DWT	Discrete Wavelet Transform
EEG	Electroencephalography
ELM	Extreme Learning Machine
FFT	Fast Fourier Transform
FN	False Negative
FP	False Positive
FPGA	Field Programmable Gate Array
FPU	Floating Point Unit
f_s	Sampling Frequency

FSA	Flux Signature Analysis
FSX	Feature Set (number) X
FT	Fourier Transform
HI	High Impedance
HOS	Higher Order Statistics
IM	Induction Motor
ISR	Interrupt Service Routine
KNN	K Nearest Neighbors
LI	Low Impedance
MAC	Multiply Accumulate
MCSA	Motor Current Signature Analysis
MCU	Microcontrollers
ML	Machine Learning
MLP	Multilayer Perceptron
MRA	Multiresolution Analysis
MST	Minimum Spanning Tree
OPF	Optimum Path Forest
OVO	One Versus One
OVR	One Versus Rest
PCA	Principal Components Analysis
RAM	Random Access Memory
RBF	Radial Basis Function
RBFNN	Radial Basis Function Neural Network
RISC	Reduced Instruction Set Computer
RMS	Root Mean Squared
RWE	Relative Wavelet Energy

SIMD	Single Instruction Multiple Data
SOM	Self Organizing Maps
STFT	Short-Time Fourier Transform
SV	Support Vectors
SVM	Support Vector Machine
TAT	Total Acquisition Time
TP	True Positive
TN	True Negative
WT	Wavelet Transform

List of symbols

n_s	Synchronous speed
f_e	Driving Frequency
s	Slip
i	Current
μ	Mean
σ	Standard deviation
ψ	Wavelet function
$\mathbb{R}$	Real numbers set
$\inf$	Infinity
s	Scaling factor
u	Translating factor
W	The wavelet transform
Lf	Low frequency approximation of function f
Hf	High frequency approximation of function f
η	Learning rate
σ^2	Variance
σ_{xk}	Covariance
Σ	Covariance Matrix
ϕ	Activation Function
α	Lagrange multipliers
C	Reciprocal of the regularization parameter in SVM
$\in$	Belong
f_c	Cutoff Frequency

Contents

1	INTRODUCTION	31
1.1	Motivation	33
1.2	Problem description	34
1.3	Objectives	34
1.4	Scientific production	34
1.5	Thesis structure	35
2	LITERATURE STUDY	37
2.1	Squirrel cage induction motor	37
2.1.1	Stator winding short-circuit failure	37
2.1.2	Stator winding short-circuit failure detection	39
2.2	Signal processing and feature extraction	41
2.2.1	Statistical features	41
2.2.2	The wavelet transform	43
2.3	Feature selection	45
2.3.1	ANOVA for feature selection	46
2.4	Machine learning techniques	47
2.4.1	Perceptron	48
2.4.2	Bayesian classifiers	49
2.4.3	Multilayer perceptron	51
2.4.4	Extreme Learning Machine	54
2.4.5	Radial Basis Function Neural Network	54
2.4.6	K-nearest neighbors	57
2.4.7	Support Vector Machine	58
2.4.8	Optimum Path Forest	63
2.5	Embedded systems	65
2.5.1	Embedded digital signal processing	66
2.5.2	Embedded machine learning	67
2.5.3	Embedded induction motor failure detection	69
3	DESIGN METHODOLOGY AND DATA ACQUISITION	71
3.1	General view	71
3.2	Data acquisition	73
3.2.1	The test bench	73
3.2.2	Dataset composition	76
3.3	Conclusion	79

4	TIME DOMAIN ANALYSIS	81
4.1	Short-circuit failure and leakage flux	81
4.2	Pattern classification	82
4.2.1	Perceptron	83
4.2.2	Bayesian classifiers	84
4.2.3	Other classifiers	85
4.2.4	Relevance of MeanAD and MedianAD	87
4.3	Evaluation of frequency and load as additional features	89
4.4	Conclusion	93
5	THE INCLUSION OF MULTIREOLUTION DECOMPOSITION IN FEATURES EXTRACTION	95
5.1	Wavelet Transform applicability and its advantages	95
5.2	DWT initial exploration	95
5.2.1	Features set exploration using KNN	96
5.2.2	Results with other classifiers	98
5.3	Increasing the levels of decomposition	101
5.3.1	Feature set exploration using KNN	101
5.3.2	Results using other classifiers	105
5.4	Summary of features set and classifiers exploration	109
5.5	Conclusion	109
6	EMBEDDED PROTOTYPING	113
6.1	Computational resource usage reduction	113
6.1.1	2 classes approach	113
6.1.2	Total Acquisition time reduction	115
6.1.3	Sampling frequency reduction	118
6.1.4	Total acquisition time and sampling frequency reduction combined	119
6.2	Embedded application platform and model choice	121
6.3	Embedded application implementation	123
6.3.1	Embedded prototype evaluation	124
6.4	Conclusion	129
7	CONCLUSION	131
7.0.1	Future works	132
	BIBLIOGRAPHY	135

APPENDIX A – ENERGIZED POLYMERIC INSULATOR DEGRADATION DETECTOR: ANOTHER CASE STUDY 145

1 Introduction

After decades of development in embedded systems, with a dramatic shift in the processing capabilities, lower power consumption and decreased footprints, those devices become present in almost all kinds of human activities, what is called ubiquitous computing (WEISER, 1993). Another field that intensively spread and advanced in the last decades is artificial intelligence (AI) techniques. This area has achieved high visibility and it is intensively applied in engineering and daily activities, often unnoticed.

The parallel progress of both areas allows the possibility of moving machine learning (ML) from general purpose computers or cloud-based data centers to edge, with smart devices becoming an intrinsic part of life. The combination of machine learning and embedded systems impacts virtually every market. It leads to changes in connecting car technologies, city infrastructure, health care, farming and manufacturing process, making the latter two far more efficient (JäGARE, 2019).

In this context, training phase of the ML system is typically performed with power and compute-intensive systems, like server systems or general purpose computers and the inference, or prediction phase, can be done locally or in the edge device, considering a wireless sensor network. While in the standard approach, the data is sent to the cloud and the prediction is also executed there. The improvements in ML techniques and the increased computational capabilities of edge devices is contributing to change this paradigm, which have the following advantages: (1) reduced latency and increased reliability, (2) save the power and cost necessary to transmit data around systems (LEE; SPANIAS; TEPEDELENLIOGLU, 2017) or even (3) make the application independent of network connections, (4) data privacy and security (JäGARE, 2019).

Despite some challenges of embedded computing are well known in the general computing world, others are specific of this field, like power consumption, deadline-driven programming and reduced memory capacity. Many embedded computers are battery powered or have power supply capacity limited by weight, cost or noise. Some applications have response time requirements that are harder to achieve with embedded systems (WOLF, 2008). However, with careful analysis of application requirements, platform capabilities, algorithms and code, high standard results are possible.

Many proposed machine learning solutions are developed without considering the real application context or that it is probably going to be executed by an embedded system. Thus, some aspects are neglected, like signal conditioning, acquisition and pré-processing or the computational cost of algorithms. This can cause many difficulties during the embedded prototyping phase, like the usage of features or signals difficult or expensive

to acquire, algorithms with high memory footprint or time complexity, making it very hard to achieve cost, response time, size or power consumption requirements. Considering those aspects from the beginning of the development should help to guide decision making and trade-offs during the project. Sometimes, much attention is given to optimize the classifier or regression stage and the signal processing phase is the one that consumes more computational resources or impacts more in execution time.

Based on what was exposed, it is presented in this thesis, the initial development of an online system to detect stator winding incipient short-circuit defects in three-phase induction motors using an axial flux signal. The simulations and model training are performed in a general purpose computer, but the prediction phase, signal acquisition, processing and classification should be executed in the embedded system. The motor should be constantly monitored and when the failure is detected, this module should set an alarm up or even turn the machine off.

The three-phase induction motor (IM) is the most common motor used in industrial applications, particularly the squirrel-cage type. Failures of those machines can lead to severe economic repercussions due to maintenance cost, manufacturing shutdown and other less tangible problems, like customer deliver delays, accidents, product quality and efficiency drop (RIERA-GUASP; ANTONINO-DAVIU; CAPOLINO, 2015). Based on this, incipient failure detection in induction motor is a promising and intense field of study.

There are three main types of failures in IM: Rotor, stator winding and mechanical faults. The second type of fault accounts from 21% (BONNETT; YUNG, 2008) to 40% (LAHOUD et al., 2013) of IM failures. Despite stator winding design and manufacturing have been improved and stator electrical failures have decreased, for machines driven by power converters, the winding is stressed out by the high harmonic content in the voltage (BELLINI et al., 2008). As this driving option has become the standard, it is important to investigate efficient ways to detect stator winding faults. Instead of periodic inspections, the idea is to develop an online diagnostic system, that can disconnect the machine before a complete failure.

The stator winding failures are usually of two types: asymmetry in the stator winding, like open-phase, or inter-turn short-circuit. The last one can cause catastrophic failure in a short time. This defect is one of the most difficult to detect incipiently and meanwhile not detected, the heat in shorted turns can lead to critical insulation breakdown. The main issue to short-circuit detection is the short time between an incipient defect, including only a few turns, and a severe failure, that can take only a few seconds. This way, a short-circuit detection system should be able to perform its analysis and shut down the defective machine in a few seconds.

1.1 Motivation

There are many real problems that can be addressed by the combination of signal processing and computational intelligence. Depending on the application, the proposed algorithm should run in an embedded system. One example is stator winding short-circuit failure online detection. Despite the diversity of possibilities already described in many papers, that includes different types of signals, many signal processing techniques and classifiers, most of the studies do not take into account the requirements for an online embedded application. It usually implies handling aspects like signal conditioning and acquisition, efficient memory usage and, specifically, for short-circuit detection, short response time. Furthermore, to make the solution viable for low power machine applications, there is another constraint: cost.

With the increase of computational power and the decrease in cost of embedded systems, developing a solution with the described characteristics became more tangible. This way, from the embedded software engineering point of view, the main motivation of this work is to explore a methodology to efficiently develop an embedded application that includes the combination of signal processing and classification tasks. Despite the study to define the algorithm is conducted on a platform different from the target, it is important to consider the specificities of embedded systems and the environment of the application. It makes the embedded prototyping easier and more efficient. It is expected that the methodology used for the development of short-circuit detection system can be applied to similar problems.

The industry 4.0 motivation, sensor, communication and information processing development made the real-time online monitoring of motors a real possibility. The latest trend in motor maintenance is the condition-based maintenance (CBM) (KOONS-STAPF, 2015), based on the evidence of need. It contrasts the typical maintenance strategies of "run-to-failure", time based or corrective/preventive maintenance, that can be costly in different ways. CBM seeks to reduce operating costs and increase safety. In this context, the stator winding short-circuit is the second most common type of induction motor failure, behind bearing failures only. This way, contributing to the development of low cost online detection system is very important to prevent or minimize damage in industrial activities. Presenting a solution capable of detecting the failures in an initial stage and efficient in computational resources usage and execution time is an important step in the development of a condition monitoring. It can be connected to the frequency converter that drives the motor, or even integrated to its hardware.

Besides, induction motor failure detection field is one of the main topics studied in Energy Control and Processing research group of the Federal Institute of Ceará and this study represents a continuation of the studies previously developed by them. In the work performed by Oliveira, Silva and Medeiros (2013) and Vieira et al. (2016), the motor

current signature analysis (MCSA) and MLP is used to detect the same type of failure and an embedded prototype is also presented. This study has similar goals but with a different approach to overcome or minimize some drawbacks of the solution proposed by them.

1.2 Problem description

Short-circuit failure is one of the most common problems in induction motor and due its wide application in industry, detecting incipiently this type of defect is a relevant field of study. The case of study addressed in this investigation is the development of an embedded incipient failure detection prototype using the axial flux signal. The online diagnostic should take a few seconds maximum, because this type of failure might present a very fast evolution in some cases. The developed solution should have low computational resources requirements, to allow the usage of low cost embedded systems, as a viable solution to low power motors. It should even possibly be integrated with frequency converter equipment (inverter). Besides, it should ideally not provide false alarms. Because it would cause unnecessary stops of the machines and also negatively affect the trust in the system, that could not be taken seriously when an alarm was issued as a result of a real fault.

1.3 Objectives

Develop a methodology for the implementation of embedded applications combining signal processing and machine learning techniques. It should be applied in the detection of stator winding incipient short-circuit in three-phase induction motors. The requirements and limitations imposed by a low cost embedded computer should be considered.

The specific objectives are:

- Evaluate feature extraction techniques to extract and select characteristics that represents well the short-circuit failure problem;
- Evaluate machine learning techniques in short-circuit detection, considering accuracy, performance and also computational resources use;
- Optimize the solution to reduce the detection computational cost and response time;
- Evaluate the embedded implementation of the solution defined in the exploration.

1.4 Scientific production

During the development of this thesis, the following publications were produced:

- Rebeca G. C. Cunha, Elias T. da Silva Jr, Joao W. M. De Souza and Pedro P. Rebouças Filho. Toward an embedded application to identify degradation in energized polymeric insulators using discrete wavelet transform. In: 2018 25th International Conference on Systems, Signals and Image Processing (IWSSIP). [S.l.]: IEEE, 2018.
- Rebeca G. C. Cunha, Elias T. da Silva Jr and Claudio M. S. Medeiros. An embedded application to identify degradation in energized polymeric insulators using machine learning and wavelet transform. In: 2018 VIII Brazilian Symposium on Computing Systems Engineering (SBESC). [S.l.]: IEEE, 2018.

1.5 Thesis structure

The text is organized as follows. Chapter 2 presents a brief explanation about the topics whose understanding is fundamental to the experiments comprehension. It includes the working principle of squirrel-cage induction motors, its applications and main defects, with a focus on the short-circuit stator winding. Some studies performed in this field are also discussed. Then, some concepts and techniques about features extraction and selection are described, including higher order statistics, the discrete wavelet transform and multiresolution decomposition. The classifiers applied are also shortly explained. By last, some considerations regarding embedded applications are presented.

In the following chapters, the methods applied and results achieved throughout the research development are described. In chapter 3, the methodology applied to the development of the failure detection prototype is presented and the dataset generation process is described. In chapter 4, the initial studies about feature extraction from the time domain signal and classification are presented. Chapter 5 describes the inclusion of the discrete wavelet transform during the signal pre-processing phase and the outcomes of this approach in the classification performance. In chapter 6, the process to decrease the computational resources usage and the embedded implementation of the solution proposed in the previous chapter is described. At last, in chapter 7, the conclusions, contributions of the study and future work possibilities are discussed.

2 Literature Study

In this chapter are explained the topics necessary to the adequate comprehension of the experiments performed in this work. The text is succinct and limited to the most relevant concepts, based on classical references and is also presented the state of art of each area.

2.1 Squirrel cage induction motor

The stator circuit is composed of three coils, that should be fed by three phase voltages lagged by 120 electrical angles each other. The alternated current is supplied directly to the stator, magnetizing it and generating a rotary field that turns with synchronous speed n_s (ROCKWELL AUTOMATION, 1996), determined by the stator poles number and the applied frequency f_e (FITZGERALD; KINGSLEY; UMANS, 2003). There is no electrical connection between stator and rotor.

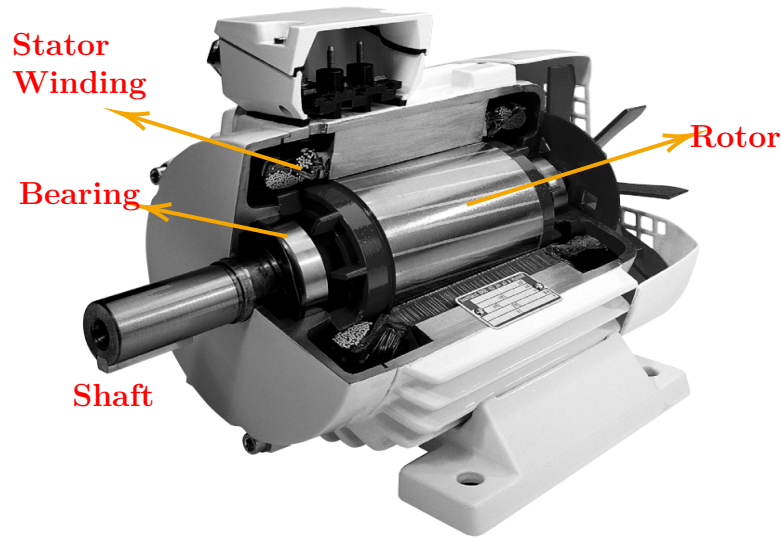
There are two types of polyphasic induction machine rotor: wound rotor and squirrel cage. The second type is the most common and is the one addressed in this work. Thus, only this kind will be discussed. The main reason for its popularity is the design simplicity and robustness (FITZGERALD; KINGSLEY; UMANS, 2003). It consists of a slotted cylindrical rotor core with aluminum bars, that are joined at the front and back by rings, to form a closed cage. Because it is shorted, its internal rotor current is dependent on the induced voltage and its impedance (ROCKWELL AUTOMATION, 1996). A squirrel cage induction motor picture is presented in Figure 1.

The interaction between the magnetic flux generated by the stator winding and the rotor bars induces a rotor current, similar to a transformer operation (FITZGERALD; KINGSLEY; UMANS, 2003). The rotor tries to rotate at the same speed as the rotating field, but it never achieves it. For this reason it is called asynchronous motor. The difference between the synchronous speed and rotor speed is called slip s . It is positive when the machine is working as motor and negative, as a generator.

2.1.1 Stator winding short-circuit failure

Loss of insulation in stator winding is one of the most common failures in induction motors. It accounts for about 21 % to 40 % of all failures in electrical machines, depending on type and size. Winding failures are usually associated with slow and continuous aging of the insulation (RIERA-GUASP; ANTONINO-DAVIU; CAPOLINO, 2015). When aged, local failures can start due to small stress, like overheating, vibration or transient voltages

Figure 1 – Three-phase squirrel-cage motor



Source: Adapted from (ROCKWELL AUTOMATION, 1996)

(BONNETT; SOUKUP, 1992). It causes an inter-turn short-circuit and current circulates through shorted turns, generating heat, increasing the insulation damage and propagating the failure to other close turns.

The evolution of the failure can lead to phase-ground and phase-phase short-circuit, with catastrophic consequences (RIERA-GUASP; ANTONINO-DAVIU; CAPOLINO, 2015). The evolution of this fault can be very fast, just a few seconds, what requires online and constant monitoring (CABANAS et al., 2013), depending on the importance and size of the machine to enable the interruption of operation before an imminent catastrophic fault. To be of practical interest, inter-turn short-circuit detection methods should be able to detect the fault very quickly, in incipient stages, when the fault current still low and before conventional protection systems like over-current and differential relays act (RIERA-GUASP; ANTONINO-DAVIU; CAPOLINO, 2015).

Another aspect that increases the importance of stator winding short-circuit detection is the predominance of frequency converters to drive the induction motors. Studies show they increase the stress in the winding because of voltage transients (BONNETT, 1996). Furthermore, Frosini, Zanazzo and Albin (2016) pointed out that the presence of high variability in the frequency components amplitude in the signals of stator current and leakage flux of inverter-fed motors, acquired at the same conditions, makes the detection harder, increasing the chances of false positives while detecting early stage inter-turns short-circuit failures.

Considering the effects of the voltage converters in the insulators of electrical machines, an international standard has been created, the IEC 60034-18-41, that describes

qualification and quality control tests for the approval of electrical insulation systems for usage in rotating electrical machines rated at 700 V_{rms} or less, driven from voltage converters. The existence of such a standard emphasizes the negative impact of voltage converters on motor insulation.

2.1.2 Stator winding short-circuit failure detection

The most common diagnostic procedures can be classified into three classes: model, signal and data based. Signal processing takes participation in all of them but with different impacts. Model-based relies on a theoretical analysis of the machine, whose model is used to predict signatures. The difference between measured and simulated signatures is used as a fault detector. In signal-based diagnostics, the known fault signatures are sought in signals sampled from real machines. Typically frequency analysis is used. Data-based diagnosis does not require knowledge of machine parameters and models. It combines signal processing and clustering techniques. Data is acquired from the machine and processed to extract features that can be used to classify them. In this type of diagnosis, AI and pattern recognition techniques are widely used (BELLINI et al., 2008).

Different kind of signals are used in signal and data driven approaches. A very common option is Motor Current Signature Analysis (MCSA), that considers the frequency components present in stator current. It is extensively analyzed in literature and there are industrial case histories (THOMSON; GILMORE, 2003). Studies performed by Tavner and Penman (1987) and Thomson (1984) proved that rotating flux waves in defective motors induce corresponding frequency components in stator current. However, one issue rarely addressed is that the usage of conventional FFT to detect fault components, implies high frequency resolution, what requires long sampling periods, which could be longer than the time to the incipient failure become catastrophic (RIERA-GUASP; ANTONINO-DAVIU; CAPOLINO, 2015).

Current measurements are widely used because they are non invasive and avoid using specific sensors. However, using leakage flux can be also a very promising option. Yazidi et al. (2005) suggested that flux signature analysis (FSA) is more effective when the number of shorted turns is small, than MCSA. Furthermore, there are many possible places to attach the sensing probe around the machine, bringing the opportunity to develop methods for accurate location of the fault (J.PENMAN, 1994) and without requiring previous knowledge of the healthy state of the machine if using two flux probes (RIERA-GUASP; ANTONINO-DAVIU; CAPOLINO, 2015). Some authors measure flux in radial direction (Zheng Liu et al., 2016), (SURYA et al., 2017), meanwhile others in axial (FROSINI; ZANAZZO; ALBINI, 2016), (J.PENMAN, 1994).

For the ideal machine, the axial leakage flux is zero, because the stator and rotor currents should be perfectly balanced. However, in practice there are asymmetries

in winding geometry and uniformity in the material. This way, there is a small, but measurable axial leakage flux in real machines. Faults represents large asymmetries in machine winding, enhancing the phenomenon of axial leakage field (J.PENMAN, 1994).

To address the disadvantages of FFT and with the knowledge that only some frequency ranges are needed, zoom-FFT¹ and the chirp Z-transform² has been developed, with the benefits of reducing computational time, memory footprint and high accuracy in the specified frequency range (BELLINI et al., 2008). Short-time Fourier transform (STFT) and statistical methods are also used.

Time-domain analysis is also applied to detect fault conditions in induction motors. Its advantages are lower computational cost and reduced time acquisition period. Many time-domain studies use starting signal transients. Signal processing techniques, like Wavelet and Hilbert-Huang transforms are used to perform a time-frequency analysis. Cusidó et al. (2008) use the wavelet transform to decompose the stator current in frequency bands and calculates the power spectral density of the detail coefficients to highlights differences between motor in normal condition, under broken bars and shorted turns conditions using transient signals. Frosini, Zanazzo and Albini (2016) use the wavelet decomposition to remove high variability of harmonics in stator current and leakage flux of inverter-fed induction motors.

The need of efficient and reliable diagnostic systems leaded to the use of computational intelligence techniques. In the diagnostic of electrical machines, neural networks are used to perform recognition and classification of the machine state. Self-organizing maps (SOM) can be used to cluster normal, rotor broken bars and shorted-turns motor patterns (WOLKIEWICZ; SKOWRON; KOWALSKI, 2018). Jawadekar et al. (2014) use current signal, the continuous wavelet transform and neural network with one hidden layer, to distinguish between healthy, bearing fault, inter-turn short-circuit and rotor bar crack, with 100 % accuracy. Rebouças Filho et al. (2018) used MCSA and multilayer perceptron to detect inter-turn short-circuit from induction generators also with 100 % accuracy, including very incipient faults with 1.41 % of shorted turns. The additional features used were: driving frequency, inverter DC-bus voltage and current RMS value.

Oliveira, Silva and Medeiros (2013) use only one current sensor to identify this kind of failure in induction motor, also including very incipient scenario, only 1.4 % of shorted turns. In their test experiments, 67 % of the failure conditions were detected. In this work, MCSA and neural network were combined. In another related work, Vieira et al.

¹ It is possible to examine a restricted frequency band in much greater detail without increasing the FFT length. It can be achieved by bandpass filter and downsampling operations (MULGREW; GRANT; THOMPSON, 2003)

² The DFT can be considered as a evaluation of the z-transform along the unit circuference $exp(jw)$. The chirp z -transform evaluates the z-transform along a general spiral contour in the z-plane (GIRON-SIERRA, 2017)

(2016) developed an embedded prototype to detect this type of failure as a continuation of Oliveira, Silva and Medeiros (2013)'s work.

The last years, deep learning is becoming very popular in many applications and induction motor failure detection has already been investigated using those techniques. Meanwhile manually defining the features has been the standard option, with deep learning algorithms, features extraction and classification tasks are combined and automatized. The deep neural networks contain multiple hidden layers to learn information from the input. Shao et al. (2017) developed a fault diagnosis approach, that uses the frequency distribution of vibration signal, after performing FFT, as input to a deep belief network, composed of stacked multiple units of restricted Boltzmann machine. Six different working states have been considered: normal, stator winding defect, unbalanced rotor, defective bearing, broken bar and bowed rotor, with 99.98 % accuracy. In Khan, Alekhya and Seshadrinath (2018), CNN (Convolutional Neural Network) and LSTM (Long Short Term Model) are applied to MCSA to detect an incipient inter-turn faults.

Considering embedded applications, despite the good results, deep learning presents several challenges. Deep models have often millions of parameters, what is a huge impact or even become unfeasible under constrained devices due to limited memory, computational power and large inference time. There are ways to minimize these problems, like parallelism, reduction in parameters precision and weights quantization (LANE et al., 2017). However, it increases the complexity of the problem and also increases the cost.

2.2 Signal processing and feature extraction

Feature extraction consists of identifying characteristics that can be used to represent the signal and relate it to an specific problem. This is a very important step in machine learning problems, because without good features is not possible to obtain good classification or regression results. In this section, the features extraction methods used in the experiments are described.

2.2.1 Statistical features

Statistical features can be extracted to reflect the statistical properties of a signal. The summary statistics is intended to communicate a great amount of information as simple as possible. It is composed of: measures of central tendency, like mean, median and mode; dispersion, like standard deviation, variance and mean absolute deviation; and shape, like skewness and kurtosis.

Skewness is the third-order statistical moment, normalized by the third power of standard deviation. It is a measure of symmetry. For a normal distribution, the skewness is zero. Negative values indicate the data is skewed left and positive values indicate it is

skewed right (SUGIYAMA, 2016). The formula used to calculate skewness in this study is the Fisher-Pearson coefficient of skewness

$$skewness = \frac{E(X_\mu)^3}{\sigma^3} = \frac{\sum_{i=1}^N (X_i - \mu)^3 / N}{\sigma^3}, \quad (2.1)$$

which μ is the mean, σ is the standard deviation and $E(X_\mu)^3$ represents the third moment of X .

Kurtosis is the fourth-order moment, normalized by the fourth power of standard deviation. It measures whether the data are heavily-tailed or lightly-tailed compared to a normal distribution (SUGIYAMA, 2016). For a normal distribution, $kurtosis = 3$. In this work, is used the "excess kurtosis"

$$kurtosis = \frac{E(X_\mu)^4}{\sigma^4} = \frac{\sum_{i=1}^N (X_i - \mu)^4 / N}{\sigma^4} - 3. \quad (2.2)$$

This way, positive kurtosis indicate heavy-tailed distribution and negative, light-tailed.

The skewness and kurtosis are part of higher-order statistics (HOS), that are extensions of second-order measures to higher orders (MCLAUGHLIN; STOGIOGLOU; FACKRELL, 1995). HOS are applicable in non-Gaussian process and many real-word application are truly non-Gaussian, but were threated on past as Gaussian based on assumptions about the nature of the signal and noise (Kletter; Messer, 1990). Besides, many measurement noises are Gaussian, and so, in principle, the HOS are less affected by Gaussian background noise than the 2nd order measures.

The applicability of HOS to machine failure detection was shown in studies performed by Martin and Honarvar (1995) and SAMANTA and AL-BALUSHI (2003), both using vibration time-domain signals to detect bearing failure in rotating machines.

Additionally to skewness and kurtosis, mean absolute deviation

$$MeanAD = mean(|X_i - X_\mu|) \quad (2.3)$$

and median absolute deviation

$$MedianAD = median(|X_i - \widetilde{X}|) \quad (2.4)$$

were also used. The first one is the average of the absolute differences between each score and the overall mean (X_μ). Its computational advantage compared to standard deviation is the elimination of the square and squared-root operations. The second is the median of the absolute differences between each score and the overall median ($\widetilde{X}$). Despite also being a dispersion measure, it is a robust statistic measure, being more resilient to outliers in a dataset than the standard deviation.

2.2.2 The wavelet transform

Wavelet analysis is a time-frequency technique. Similarly to the Fourier Transform, it decomposes signals over oscillatory waveforms, revealing signal properties. However, the difference between them is that the Fourier Transform represents any infinite energy function as a sum of sinusoidal waves, with linear time-invariant operators, for uniformly regular signals. Meanwhile a wavelet basis defines a sparse representation of piece-wise regular signals, which may include transient and singularities. In images, large wavelet coefficients are located in the neighborhood of edges and irregular textures (MALLAT, 2009).

Wavelet transforms are based on small waves with limited duration, generated in the form of translation and dilation of a fixed function, called mother Wavelet . Those waves have special scaling properties, permitting a closer connection between the function being represented and their coefficients (the result of the wavelet transform). There are many applications for the wavelet transform, including signal processing, data compression and harmonic analysis. Every application using FFT can be formulated using wavelets, providing more localized temporal (or spatial) and frequency information (LEE; YAMAMOTO, 1994).

A wavelet is a function $\psi \in L^2(\mathbb{R})$, with zero average (MALLAT, 2009):

$$\int_{-\infty}^{+\infty} \psi(t) dt = 0. \quad (2.5)$$

It is normalized $|\psi| = 1$ and centered in the $t = 0$. It can be scaled and translated by s and u , respectively

$$\psi_{u,s}(t) = \frac{1}{\sqrt{s}} \psi\left(\frac{t-u}{s}\right), \quad (2.6)$$

which $u \in \mathbb{R}$ and $s \in \mathbb{R}^+$. ψ is called the mother wavelet and $\psi_{u,s}$, the wavelet basis.

The wavelet transform of $f \in L^2(\mathbb{R})$ at time u and scale s is

$$Wf(u, s) = \langle f, \psi_{u,s} \rangle = \int_{-\infty}^{+\infty} f(t) \frac{1}{\sqrt{s}} \psi\left(\frac{t-u}{s}\right) dt. \quad (2.7)$$

The scaling function ϕ_s is an aggregation of wavelets at scales larger than 1, it is

$$\phi_s(t) = \frac{1}{\sqrt{s}} \phi\left(\frac{t}{s}\right) \quad (2.8)$$

and $\bar{\phi}_s(t) = \phi_s^*(-t)$. It is used to obtain the low-frequency approximation of $f[t]$

2.2.2.1 Discrete wavelet transform

Let $f(t)$ be a continuous time signal defined over $[0,1]$ and $f[n]$ its discrete version. Its discrete wavelet transform can be calculated for $s = a^j$, with $a = 2^{\frac{1}{v}}$, providing v

intermediate scale. Let $\psi(t)$ be a wavelet, its discrete version scaled by a^j is (MALLAT, 2009). Thus,

$$\psi_j[n] = \frac{1}{\sqrt{a^j}} \psi\left(\frac{n}{a^j}\right) \quad (2.9)$$

and

$$Wf[n, a^j] = \sum_{m=0}^{N-1} f[m] \psi_j^*[m - n], \quad (2.10)$$

which $f[n]$ and $\psi_j[n]$ are considered periodic signals with period N . And $\bar{\psi}_j[n] = \psi_j^*[-n]$.

To add the low frequencies $Lf[n, a^j]$, for scales larger than a^j , a discrete and periodic scaling filter is computed by sampling the function $\phi(t)$, which

$$\phi_J[n] = \frac{1}{\sqrt{a^j}} \phi\left(\frac{n}{a^j}\right) \quad (2.11)$$

and

$$Lf[n, a^j] = \sum_{m=0}^{N-1} f[m] \phi_J^*[m - n], \quad (2.12)$$

where $\bar{\phi}_J[n] = \phi_J^*[-n]$.

2.2.2.2 Wavelet analysis and multi-resolution decomposition

The wavelet transform can be used for multi-scale analysis of a signal through dilation and translation, so it is able to extract time–frequency features of a signal effectively. Due to the multi-scale analysis ability, the wavelet transform is able to produce high frequency resolution at low frequencies and high time resolution at high frequencies. The wavelet analysis provide a signal decomposition in independent frequency bands (LEI, 2017). Decomposition coefficients are computed with a fast algorithm that cascades discrete convolutions with h , and g , and downsamples the output (MALLAT, 2009). It decomposes successively each approximation coefficient a into a coarser approximation level, plus the wavelet coefficients carried by the details d . What can be analytically expressed as

$$a_{j+1}[p] = \sum_{n=-\infty}^{+\infty} h[n - 2p] a_j[n] = a_j \star \bar{h}[2p] \quad (2.13)$$

and

$$d_{j+1}[p] = \sum_{n=-\infty}^{+\infty} g[n - 2p] a_j[n] = a_j \star \bar{g}[2p], \quad (2.14)$$

where $2p$ represents the even index samples that are keep after sub-sampling and h and g are defined below (MALLAT, 1989):

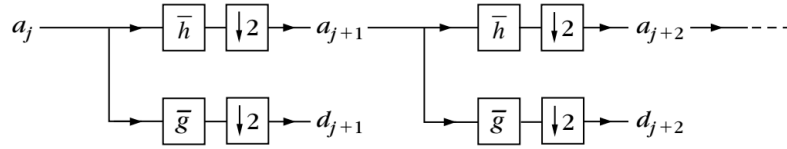
$$g[n] = \left\langle \frac{1}{\sqrt{2}} \psi\left(\frac{t}{2}\right), \phi(t - n) \right\rangle \quad (2.15)$$

and

$$h[n] = \left\langle \frac{1}{\sqrt{2}} \phi\left(\frac{t}{2}\right), \phi(t - n) \right\rangle. \quad (2.16)$$

In Figure 2, there is a diagram representing the wavelet Multi-resolution Analysis (MRA). The filter $\bar{h}$ is a low pass filter that removes the higher frequencies of a_j , whereas $\bar{g}$ is a high-pass filter that collects the highest frequencies. The block $\downarrow 2$ represents the sub-sampling (decimation) operation.

Figure 2 – Wavelet decomposition computed using the fast wavelet transform, with cascade of filterings $\bar{h}$ and $\bar{g}$



Source: Mallat (2009)

2.2.2.2.1 Relative wavelet energy

Considering that the condition of orthonormality is maintained, Parseval's theorem relates the energy of the signal $x(t)$ to the energy in each of the components and their wavelet coefficients. The energy of the detail signal at resolution level j is

$$E_j = \sum_k |d_{j,k}|^2. \quad (2.17)$$

The relative wavelet energy (RWE) at resolution j is, thus

$$RWE_j = \frac{E_j}{E_{total}}, \quad (2.18)$$

which

$$E_{total} = \sum_{j=1}^{N+1} E_j, \quad (2.19)$$

which N is the number of levels of decomposition. This method accurately characterizes the specific phenomenon related to the different frequency bands in time and frequency planes. This feature is often used for EEG applications (GARG et al., 2011). It represents the energy percentage of the original signal, contained in that frequency band.

2.3 Feature selection

Meanwhile feature extraction is the process of finding useful characteristics in raw data, feature selection is related to reduce the amount of characteristics used to represent the problem, keeping the most relevant of them. It can be motivated by different aspects like performance improvement, saving storage requirements or increase algorithm speed.

There are mainly two types of feature selection methods: filter and wrapper. The first one is usually related to ranking the features individually based on some criteria. The second, a search is performed in nested subsets of the feature set, using a particular predictor to evaluate the performance. Also, there are hybrid methods, that rank the features using a filter and based on the ranking order, nested subsets of features are generated and their performance is evaluated using machine learning techniques (GUYON; ELISSEEFF, 2006). This is the approach used in feature selection stage of this work. A third method type, called Embedded methods, do not separate the learning from the feature selection.

The feature filter is a function returning a relevance index $J(S|D)$ that estimates, given the data D , how relevant a given feature subset S is for the task Y (usually classification or approximation of the data). Relevance indexes may also be computed for individual features $X_i, i = 1...N$, establishing a ranking order $J(X_1) \leq J(X_2) \dots \leq J(X_N)$. The features with the lowest ranks are removed (DUCH, 2006).

In the case of filter methods, there is no direct dependence of the relevance index on the predictors. To overcome this, the hybrid methods enable the evaluation of the feature contributions to the final model, similar to wrapper methods, in upwards or backwards direction, based on the filter ranking. This procedure is computationally less expensive than trying all the combinations of features, like in pure wrapper using cross-validation. Besides, some authors indicate it is also less prone to over-fitting (DUCH, 2006).

2.3.1 ANOVA for feature selection

For problems with continuous input and categorical outputs (like many classification applications), ANOVA (Analysis of Variance) can be used. The test statistic is the between-category variance and within-category variance ratio (DREYFUS; GUYON, 2006). It is an F-statistics. Thus, the score F is defined by

$$F = \frac{BCV}{WCV}, \quad (2.20)$$

where

$$BCV = \sum_{i=1}^K \frac{n_i(\bar{X}_i - \bar{X})^2}{K - 1} \quad (2.21)$$

and

$$WCV = \sum_{i=1}^K \sum_{j=1}^{n_i} \frac{(X_{ij} - \bar{X}_i)^2}{N - K} \quad (2.22)$$

are, respectively, between and within category variance. $\bar{X}_i$ denotes the sample mean in the i -th class, n_i is the number of observations in the i -th group, $\bar{X}$ is the overall mean of the data, and K is the number of categories. X_{ij} is the j -th observation in the i -th out of the K categories and N is the overall number of observations.

Based on the above expressions, the F-score is high if BCV is large relative to WCV, what means that this feature has a good discriminate capability. Hence, all features can be ranked according to their F values. In this work, after the features are ranked, features backward elimination is applied, until the classification performance has started to decrease.

2.4 Machine learning techniques

The term machine learning refers to the computer science field related to the development of computer programs that automatically improve with experience (MITCHELL, 1997). Pattern classification is one of machine learning applications. In this problems, a model should represent the relation between input characteristics and a categorical output (BISHOP, 2006). The applications are very diverse, like the detection of a diseases based on features extracted from medical images, or the prediction of a machine failure based on signals related to its functioning. Those were examples of binary classifications, but there are also multi-class problems, like detecting different types of tumor, many machine failure types, hand-written characters recognition, recognition of different objects in images, among many others.

The induction motor stator winding short-circuit detection can be addressed as a multi-class classification problem. The objective is to relate the axial flux data to the stator winding state, that can be normal or present different short-circuit failure intensities. Before briefly describing the machine learning techniques applied, some key words are defined:

- **Pattern:** A vector of features used to synthesize the information contained in raw data;
- **Label:** The category name or number;
- **Training:** The process applied to determine the parameters of the model that best represent or solve the problem (BISHOP, 2006);
- **Testing:** The verification of model capability to apply the knowledge obtained during the training phase to unseen data;
- **Hyperparameters:** The set of free parameters that controls the model complexity. They are not learned during the training phase and different combinations should evaluated to achieve the best performance on new data (BISHOP, 2006);
- **Model Accuracy** The proportion of the total number patterns correctly classified.

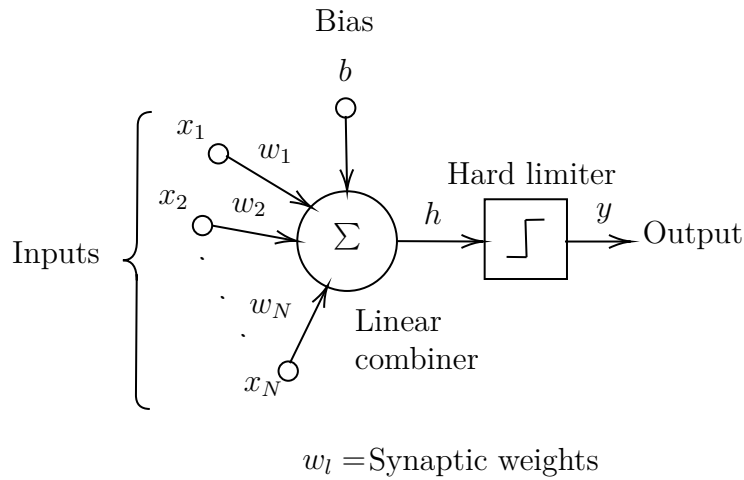
2.4.1 Perceptron

The perceptron has an important and special place in neural network development history. It was the first algorithm describing neural networks and supervised learning (HAYKIN, 2008) and was developed by Rosenblatt (1958). It is based on the McCulloch and Pitts (1943) neuron, the first model of neural networks as computing machines. The McCulloch and Pitts neuron diagram is drawn in Figure 3. Its operation consists of a linear combination of the inputs applied to the synapses, and the incorporated bias b , represented by h , followed by a hard limiter operation (sign function), which the output y is equal to +1 if h is positive and -1, otherwise. The perceptron's operation is analytically described by

$$y = \begin{cases} 1, & \text{if } h \geq 0 \\ -1, & \text{otherwise,} \end{cases} \quad \text{which} \quad (2.23)$$

$$h = \sum_{l=1}^N w_l x_l + b. \quad (2.24)$$

Figure 3 – McCulloch and Pitts neuron model representation



Source: Adapted from Haykin (2008)

The perceptron goal is to correctly classify the externally applied stimuli X into one of the two classes (1 or -1). This way, there are 2 decision regions separated by an hyper-plane, defined by

$$\sum_{l=1}^N w_l x_l + b = 0. \quad (2.25)$$

The effect of the bias b is to shift the decision boundary away from the origin. During the training phase, the synaptic weights can be adapted iteratively, based on *The Perceptron Convergence Theorem* (ROSENBLATT, 1957), that has the following learning rule

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \eta e(t) \mathbf{x}(t), \quad (2.26)$$

where η is the learning-rate parameter, $0 < \eta \leq 1$, and $e(t) = d(t) - y(t)$ is the error signal for the pattern presented on iteration t . $\mathbf{w} = [b, w_1, w_2, \dots, w_N]$, $\mathbf{x} = [x_0, x_1, x_2, \dots, x_N]$ and $d(t)$ is the label.

The perceptron only give good classification results for linearly separable problems, what is a significant limitation, considering many real problems do not fill this requisite.

2.4.2 Bayesian classifiers

The Bayesian Classifiers are based on Bayes Theorem. It attests the probability $P(c_k|X)$ that a given patten X belongs to class c_k . It is calculated as

$$P(c_k|X) = \frac{p(X|c_k)P(c_k)}{P(X)}, \quad (2.27)$$

which $p(X|c_k)$ is the density probability function of class c_k , $P(X)$ is the overall probability of an observation and $P(c_k)$ is the overall probability of given class c_k (BISHOP, 2006). The class with the highest probability ($P(c_k|X)$) is the final classification.

When the features are continuous values, it is common to assume that they follow a Gaussian distribution. If the data is D dimensional

$$p(\mathbf{X}|c_k) = \frac{1}{|\Sigma_k|^{\frac{1}{2}}(2\pi)^{\frac{D}{2}}} e^{-\frac{1}{2}(\mathbf{X}-\mu_k)^T \Sigma_k^{-1}(\mathbf{X}-\mu_k)}, \quad (2.28)$$

which Σ_k is class c_k covariance matrix, assuming each classes has a different covariance matrix. The covariance σ_{xy} measure the joint variability of two random variables x and y .

The quadratic discriminant function $g_k(x)$ is defined as

$$g_k(\mathbf{x}) = \ln(P(c_k)) - \frac{1}{2}\ln(|\Sigma_k|) - \frac{1}{2}(\mathbf{x} - \mu_k)^T \Sigma_k^{-1}(\mathbf{x} - \mu_k). \quad (2.29)$$

2.4.2.1 Alternative covariance matrices estimates

If there are few patterns per class in the training set, the use of independent covariance matrices for each class can lead to poor results, because the mean and covariance estimatives do not represent well the real problem. To minimize this, some impositions are made to Σ , as described in the following subsections.

2.4.2.1.1 General covariance matrix

With one covariance matrix for all the classes, calculated considering all the training patterns, equation 2.29 is simplified to

$$g_k(\mathbf{x}) = \mu_k^T \Sigma_{general}^{-1} \mathbf{x} + [\ln(P(c_k)) - \frac{1}{2} \mu_k^T \Sigma_{general}^{-1} \mu_k]. \quad (2.30)$$

2.4.2.1.2 Pooled covariance matrix

Another way of also using a common covariance matrix for all the classes is using a weighted combination of the independent covariance matrices Σ_k

$$\Sigma_{pool} = \sum_{k=0}^C \frac{n_k}{n} \Sigma_k, \quad (2.31)$$

which n_k is the number of patterns in class c_k , n is the total number of patterns and C is the number of classes. This way, the equation 2.29 also become a linear discriminant like equation 2.30.

2.4.2.1.3 Diagonal covariance matrix

Based on the assumption that all the features are statistically independent, the covariance matrix can be rewritten, with $\sigma_{ij} = 0$, for all $i \neq j$

$$\Sigma_{identity} = \begin{bmatrix} \sigma_1^2 & 0 & \dots & 0 \\ 0 & \sigma_2^2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \sigma_m^2 \end{bmatrix} \quad (2.32)$$

The independence between features is the assumption that defines the Naive-Bayes classifier.

2.4.2.1.4 Identity covariance matrix

A special case of diagonal matrix is the identity, which the variance for all features is equal. It results in

$$\Sigma_{identity} = \sigma^2 I_D, \quad (2.33)$$

which I_D is the identity matrix of dimension DD . This way, equation 2.29 can be rewritten as

$$g_k(x) = \frac{1}{\sigma^2} \mu_k^T x + [\ln(P(c_k)) - \frac{1}{2\sigma^2} \mu_k^T \mu_k]. \quad (2.34)$$

2.4.2.1.5 Friedman's covariance matrix

A regularization method to avoid performance decrease of quadratic discriminant for datasets with a few patterns per class has been proposed by Friedman (1989). In this approach, the k -th class estimated covariance matrix Σ_k is replaced by a matrix Σ_k^λ , originated by the linear combination of Σ_k and Σ_{pool} , that results in

$$\Sigma_k^\lambda = \frac{(1 - \lambda) \Sigma_k + \lambda \Sigma_{pool}}{(1 - \lambda) n_k + \lambda n}, \quad (2.35)$$

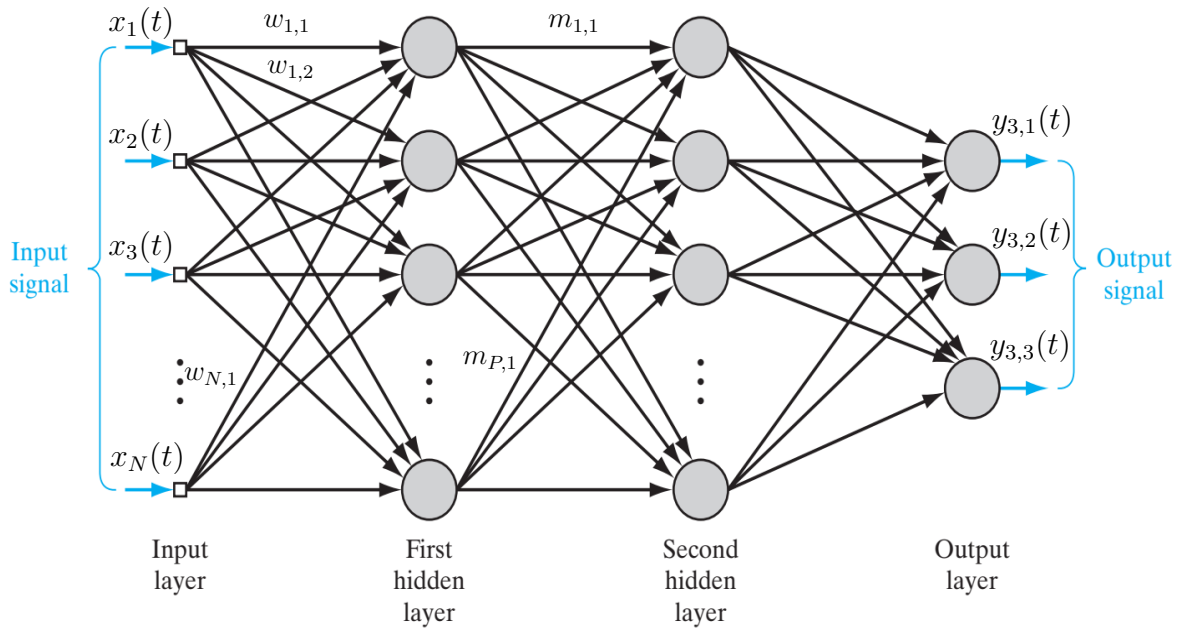
which $0 \geq \lambda \leq 1$. Thus, for $\lambda = 0$, $\Sigma_k^\lambda = \Sigma_k$ and the discriminant function is quadratic. If $\lambda = 1$, $\Sigma_k^\lambda = \Sigma_{pool}$ and discriminant is linear.

2.4.3 Multilayer perceptron

The multilayer perceptron (MLP) is a neural network with one or more hidden layers, based on Rosenblatt (1958) perceptron unit, with a differentiable nonlinear activation function φ instead of *sign* function (HAYKIN, 2008).

The MLP is a highly connected neural network, that a neuron of any layer is connected to all the neurons of previous layer. The signals flow through the network in forward direction layer by layer. The input signal $\mathbf{x}$, is fed to the input layer, that are source nodes and not perform any operation to the signal. Then, to the first hidden layer and so on for the rest of the network, until the output layer, as shown in Figure 4.

Figure 4 – Multilayer perceptron with two hidden layers



Source: Adapted from Haykin (2008)

The hidden nodes perform a nonlinear transformation, mapping the input data space into a new space called feature space (HAYKIN, 2008). In this space, it is expected that the patterns of different classes can be separated more easily.

During the prediction phase, the computations are similar to what has already been explained to perceptron. Lets call $\mathbf{x}(\mathbf{n}) = [x_1(n), x_2(n), x_3(n), \dots, x_i(n), \dots, x_N(n)]$ the input pattern n , which $x_1, x_2, \dots, x_N$ are the input features. The local induced field $h_j(n)$,

associated with neuron j is therefore

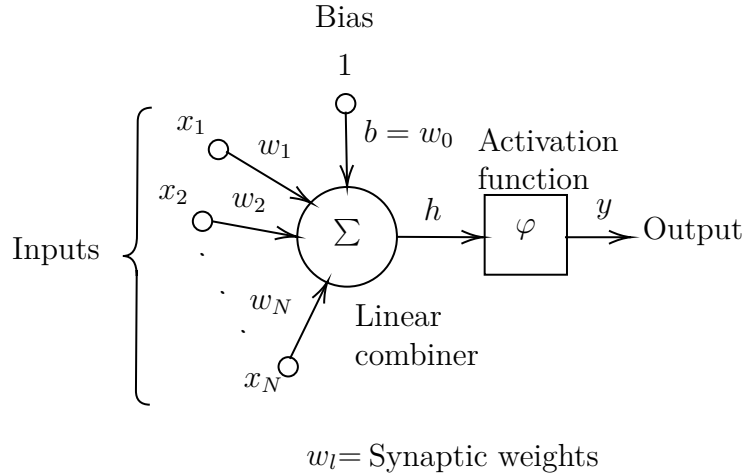
$$h_j(n) = \sum_{i=0}^N w_{ji}(n)x_i(n), \quad (2.36)$$

where N is the number of inputs. The synaptic neuron weight w_{j0} , correspond to bias b applied to neuron j . Hence, the output signal $y_{l,j}$, of layer l and neuron j is

$$y_{lj}(n) = \varphi_j(h_j(n)), \quad (2.37)$$

which $\varphi(t)$ is the activation function. In the next layer $(l + 1)$, the input vector $\mathbf{x}$ is the output vector of $\mathbf{y}_l$ previous layer. In Figure 5 there is a diagram of a neuron and its signals.

Figure 5 – Neuron representation



Source: Author

In this study, only one hidden layer is used and the activation function for all neurons is the hyperbolic tangent, defined by

$$y_i(t) = \frac{1 - \exp(-u_i(t))}{1 + \exp(-u_i(t))}. \quad (2.38)$$

Considering N the number of attributes, P the hidden units number and K the number of outputs, the number of parameters to be defined during the training phase is $L = (N + 1)P + (P + 1)K$, including the bias weights.

2.4.3.1 The back-propagation algorithm

It is a very popular algorithm for training MLPs. It can be subdivided in two phases: **forward**, similar to what was described for prediction, that the signals flow from the input layer to output layer, and **backward**, that an error signal is propagated through

the network from output to input, to perform the synaptic weights adjustment, that characterizes the learning process (HAYKIN, 2008).

Consider a MLP with 1 hidden layer, with $\mathbf{w}_j = [w_{j0}, w_{j1}, \dots, w_{ji}, \dots, w_{jN}]$ the synaptic weights associated to each hidden neuron j , and $\mathbf{m}_k = [m_{k0}, m_{k1}, \dots, m_{kP}]$ the equivalent to each output neuron k . One iteration of back-propagation algorithm follow the steps below:

1. Initialize the weight vectors $\mathbf{w}_j$ and $\mathbf{m}_k$ with random values;
2. For one training pattern $\mathbf{x}_i(n)$, with bias input append in the beginning of the vector $\mathbf{x}_i(n) = [1, x_1(n), x_2(n), \dots, x_N(n)]$, perform the calculations described in 2.36, 2.37 and 2.38 for each neuron, obtaining $h_j(n)$ and y_{1j} ;
3. Repeat item 2, for the second layer with $\mathbf{x}_j(n) = [1, y_{11}, y_{12}, \dots, y_{1P}]$, calculating $h_k(n)$ and y_{2k} ;
4. Calculate the error between the desired output $d_k(n)$ and the actual output $y_{2k}(n)$, as $e_k(n) = d_k(n) - y_{2k}(n)$;
5. Calculate the local gradient of output neurons defined by

$$\delta_k(n) = e_k(n) \varphi'_k(h_k(n)), \quad (2.39)$$

which $\varphi'_j(h_k(n))$ is the derivative of the associated activation function, hyperbolic tangent in this case. Its derivative is

$$\varphi'_j(h_k(n)) = \frac{1}{2}[1 - y_k^2(n)]. \quad (2.40)$$

6. Calculate the local gradient of hidden neurons, that is the product of the associated derivative $\varphi'_j(h_k(n))$ and the weighted sum of δ s computed for the output layer neurons, connected to neuron j , described as

$$\delta_j(n) = \varphi'_j(h_j(n)) \sum_k \delta_k(n) m_{kj}(n). \quad (2.41)$$

7. Update the output and hidden neurons synaptic weights.

$$w_{ji}(n+1) = w_{ji}(n) + \eta \delta_j(n) x_i(n), \text{ if hidden neuron,} \quad (2.42)$$

$$m_{kj}(n+1) = m_{kj}(n) + \eta \delta_k(n) x_j(n), \text{ if output neuron,} \quad (2.43)$$

where η is the learning rate.

2.4.4 Extreme Learning Machine

Huang, Zhu and Siew (2006) demonstrated the hidden neurons synaptic weights of feed-forward neural networks can be randomly assigned if the activation function of the hidden layer is infinitesimally differentiable. After that, the output weights can be analytically calculated, based on least squares solution. The ELM is a feed-forward NN with one hidden layer, with less hyper parameters for training compared to MLP and RBF, for instance.

For an arbitrary training set $(\mathbf{x}(n), \mathbf{d}(n))$, where $\mathbf{x}(n) = [x_0, x_1, x_2, \dots, x_N] \in \mathbb{R}^{N+1}$, and $\mathbf{d}(n) = [d_1, d_2, \dots, d_K] \in \mathbb{R}^K$, with P hidden nodes and activation function $\varphi(x)$, consider $\mathbf{w}_j = [w_{j0}, w_{j1}, \dots, w_{ji}, \dots, w_{jN}]^T$ the synaptic weights associated to each hidden neuron j , and $\mathbf{m}_k = [m_{k0}, m_{k1}, \dots, m_{kP}]^T$ the equivalent to each output neuron k , where w_{j0} and m_{k0} are the bias weights for hidden and output layers, respectively. The training phase consists of the following steps:

1. Randomly assign the hidden layer weights vector $\mathbf{w}_j$ for all the neurons $j = 1, \dots, P$;
2. Calculate the hidden layer output matrix

$$\mathbf{H} = \begin{bmatrix} \varphi(\mathbf{w}_1 \cdot \mathbf{x}(1)) & \dots & \varphi(\mathbf{w}_P \cdot \mathbf{x}(1)) \\ \vdots & \dots & \vdots \\ \varphi(\mathbf{w}_1 \cdot \mathbf{x}(N_{tr})) & \dots & \varphi(\mathbf{w}_P \cdot \mathbf{x}(N_{tr})) \end{bmatrix}, \quad (2.44)$$

which is equivalent to $\mathbf{H} = \mathbf{X}\mathbf{W}$, where $\mathbf{W} = [\mathbf{w}_1, \dots, \mathbf{w}_j, \dots, \mathbf{w}_P]$ and $\mathbf{X} = [\mathbf{x}(1), \dots, \mathbf{x}(N_{tr})]^T$;

3. Calculate output weights matrix

$$\mathbf{M} = \mathbf{H}^\dagger \mathbf{D}, \quad (2.45)$$

which $\mathbf{H}^\dagger$ is the Moore-Penrose generalize inverse of matrix $\mathbf{H}$ and $\mathbf{D} = [\mathbf{d}(1), \dots, \mathbf{d}(N_{tr})]^T$, considering that the output is $y_k(n) = \mathbf{m}_k \mathbf{h}(n)$.

The difference between MLP and ELM neural networks is regard the weight definition, during the training phase. The prediction procedure is the same for both, because after trained ELM is a multilayer perceptron network.

2.4.5 Radial Basis Function Neural Network

RBFNN presents structure similar to MLP, with a few differences (HAYKIN, 2008):

- The connection between input and hidden layer is not an inner product, but a distance calculation;

- The computation performed in hidden neurons is a radial basis function

$$\varphi_j(x) = \varphi(\|x - \mu_j\|), j = 1, 2, \dots, N, \quad (2.46)$$

which μ_j are the radial-basis function center, x is the input pattern and N is the number of hidden units. The Gaussian function

$$\varphi_j(x) = \exp(-\frac{1}{2\sigma_j^2}\|x - \mu_j\|^2), j = 1, 2, \dots, N \quad (2.47)$$

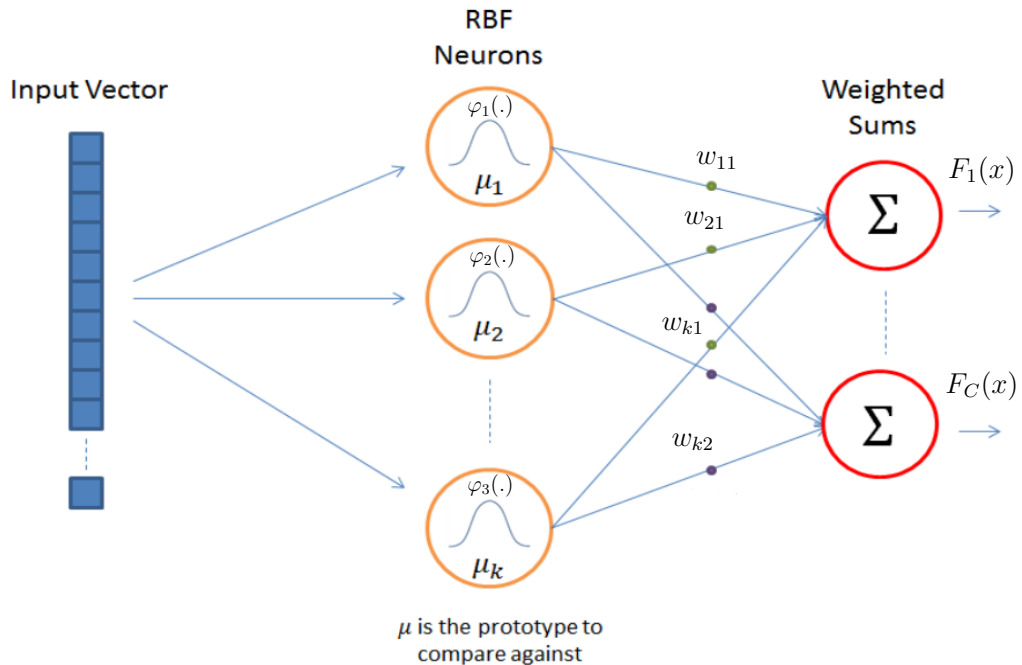
is often used. Where σ_j is the spread of Gaussian function with center μ_j . Thus, each RBF neuron compares the input vector to its prototype and outputs a value between 0 and 1, which is a measure of similarity;

- Usually, the hidden layer is trained in an unsupervised manner;
- Output layer can be linear or non linear and it is defined using supervised training.

Figure 6 presents a diagram of RBFNN, that can be analytically described by

$$F_c(\mathbf{x}) = \sum_{j=1}^K w_{jc} \varphi(\mathbf{x}(n), \mu_j). \quad (2.48)$$

Figure 6 – RBF neural network diagram



Source: Adapted from McCormick (2013)

There are many options for the definition of the RBF centers. They can be defined to be equal to one of the training set patterns, or using clustering techniques, like k-means or SOM. K-means is the most widely used technique to position the RBF centers. In the next subsections, the training phase of the RBF network applied study is briefly described.

2.4.5.1 K-means

Consider a training set composed by N pattern, that K centers should be defined, with $K < N$. The initial position of the centers are randomly assigned. At each iteration, the distance, or similarity, between each training pattern and the k centers are calculated using the Euclidian distance. Each patterns is grouped to its closest center. Then, the new μ_j center positions are defined as the center of each cluster. This process is repeated until there is no modifications in the center position and the cluster each patter is associated (HAYKIN, 2008). At the end, centers with no training patterns associated are removed.

Two different approaches for k-means had been applied: clustering all the patterns at once, class independent, in a basic unsupervised clustering, which is the most common way of using k-means; and taking advantage of the class label information and performing the clustering class by class (class-specific clustering) . This last method has the advantage of avoid different class patterns into one cluster, specially for patterns located near the decision surface, locating class uniform clusters. It usually improves the RBFNN accuracy, as demonstrated by Raitoharjuand, Kiranyaz and Gabbouj (2015), with 25 benchmark data sets used to compare class-specific k-means to other clustering methods, to the specific task of defining RBFNN centers.

2.4.5.2 Defining σ

After defining the prototypes μ_j , the spread of the Gaussian function should be specified. It can be done using grid-search, or calculate it in accordance with the spread of the centers discovered by K-means algorithm (HAYKIN, 2008).

$$\sigma = \frac{d_{max}}{\sqrt{2K}}, \quad (2.49)$$

where K is the number of prototypes and d_{max} is the maximum distance between them. This ensures the Gaussian units are not too peaked or too flat (CHERUKU; EDLA; KUPPILI, 2017), what can impact the results.

2.4.5.3 Output layer training

The definition of output weights in RBFNN can be done using least-squares estimation, similar to what is explained in Section 2.4.4, or back-propagation, as described in Section 2.4.3.1. In this dissertation, least-squares estimation was applied. Thus, after defining the centers μ_j and the width σ , the following steps are performed:

1. Calculate the activation values of RBF neurons for all the training set using equation

2.47, obtaining the matrix

$$\phi(\mathbf{X}) = \begin{bmatrix} \varphi(\mathbf{x}(1), \mu_1) & \dots & \varphi(\mathbf{x}(1), \mu_k) \\ \vdots & \dots & \vdots \\ \varphi(\mathbf{x}(N_{tr}), \mu_1) & \dots & \varphi(\mathbf{x}(N_{tr}), \mu_k) \end{bmatrix}, \quad (2.50)$$

which N_{tr} is the number of training patterns;

2. Append Bias column to first column of $\phi(\mathbf{X})$;
3. Calculate the output weights matrix

$$\mathbf{W} = \phi(\mathbf{X})^\dagger \mathbf{D}, \quad (2.51)$$

which $\phi(\mathbf{X})^\dagger$ is the Monroe-Penrose (1955) generalize inverse of matrix $\phi(\mathbf{X})$ and $\mathbf{D} = [\mathbf{d}(1), \dots, \mathbf{d}(N_{Tr})]^T$.

Training RBF can be faster than MLP, even using recursive least squares, however prediction phase is faster in MLP (HAYKIN, 2008).

2.4.6 K-nearest neighbors

KNN is an old classification method, that despite often do not achieve the highest possible accuracy when compared to more complex algorithms, due its simplicity and flexibility, it is a useful baseline against which to check more sophisticated algorithms. However, despite its simplicity, it present surprisingly good performance in many cases, for this reason KNN still a very popular classifier (LIU; MOORE; GRAY, 2006).

Based on the assumption that "samples that are close together have categories that are close together" (COVER; HART, 1967), the nearest neighbor decision rule assigns to an unclassified pattern, the classification of the nearest neighbor.

Consider a set of pairs $(\mathbf{x}_i, d_i)$, $i = 1, 2, \dots, N$, which $\mathbf{x}_i$ is the input pattern i feature vector of dimension D and d_i is the label associated. If the features values are threatred as coordinates in a dimensional space $\mathbb{R}^D$, using a metric $M(\mathbf{x}_p, \mathbf{x}_q)$, the distance between two patterns can be calculated. Thus, considering only one nearest neighbor, a new pattern $\mathbf{x}_p$ class prediction is equal to its nearest neighbor's category d_i if

$$M(x_i, x_p) = \min(M(x_i, x_p)), i = 1, 2, \dots, N. \quad (2.52)$$

The $n - 1$ remaining classifications are not considered.

If the number of samples is large, it might be a good option to use the majority vote of the nearest K-neighbors. A large K minimizes the effect of outliers, meanwhile a small K is desired in order to ensure the K-points are close enough to x_p to give an accurate estimate of its true class (COVER; HART, 1967).

2.4.7 Support Vector Machine

The support vector machine is a binary learning algorithm, proposed by Cortes and Vapnik (1995). Given a training set, it constructs a hyper-plane as the decision surface, maximizing the margin of separation between examples with positive and negative labels. New patterns are predicted to belong to a category based on the side of the hyper-plane they are located. The central idea to this algorithm development is the inner-product kernel between a support vector $\mathbf{x}_i$ and an input vector $\mathbf{x}$ (HAYKIN, 2008). The support vectors (SV) are a subset of the training set that obey some requirements explained below.

SVM can be used in linearly and non-linearly separable classification problems. To accomplish this, input vectors are non-linearly mapped to a very high dimension feature space Z and there, a linear decision surface is constructed with special properties that ensure high generalization ability (CORTES; VAPNIK, 1995).

Consider the linearly separable training set $(\mathbf{x}_i, d_i)_{i=1}^N$, where x_i is the i -th input pattern and d_i is the desired output (label). The decision surface to separate the classes is

$$\mathbf{w}^T \mathbf{x} + b = 0, \quad (2.53)$$

which $\mathbf{x}$ is the input vector, $\mathbf{w}$ is an adjustable weight vector and b is a bias. Considering the classification, we may thus write

$$\begin{aligned} \mathbf{w}^T \mathbf{x} + b &\geq 0 \text{ for } d_i = +1, \\ \mathbf{w}^T \mathbf{x} + b &< 0 \text{ for } d_i = -1. \end{aligned} \quad (2.54)$$

The optimal hyper-plane described by

$$\mathbf{w}_0^T \mathbf{x} + b_0 = 0 \quad (2.55)$$

is the unique hyper-plane that separates the training set classes with a maximal margin. It determines the direction where the distance between the projections of the training vectors of two different classes is maximal (CORTES; VAPNIK, 1995), as drawn in Figure 7. To obtain the optimal hyper-plane, the following restriction apply

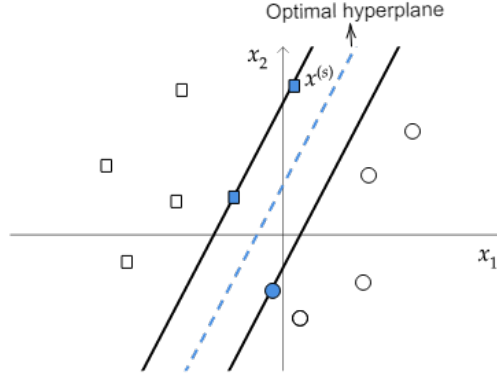
$$d_i(\mathbf{w}^T \mathbf{x} + b) \geq 1. \quad (2.56)$$

The SV are the closest points to the optimal hyper-plane and are therefore the most difficult to classify. They satisfy the equality sign of Equation 2.56.

During the training phase, the objective is to maximize the margin, by minimizing the Euclidean norm of the weigh vector $\mathbf{w}$, what is described as

$$\Phi(w) = \frac{1}{2} \mathbf{w}^T \mathbf{w} = \frac{1}{2} \|\vec{w}\|^2. \quad (2.57)$$

Figure 7 – Illustration of an optimal hyperplane for linearly separable patterns. The blue data points are the support vectors.



Source: Author

This way, the SVM training is an optimization problem, to define the weight vector $\mathbf{w}$ and bias b such that $\Phi(w)$ should be minimized and the Equation 2.56 should be satisfied. This is known as primal problem and can be solved by the method of Lagrange multipliers, using the following Lagrangian function

$$J(\mathbf{w}, b, \alpha) = \frac{1}{2} \mathbf{w}^T \mathbf{w} - \sum_{i=1}^N \alpha_i [d_i (\mathbf{w}^T \mathbf{x}_i) + b] - 1, \quad (2.58)$$

where α_i are the Lagrange multipliers.

It can be converted to a dual problem defined as

$$Q(\alpha) = J(\mathbf{w}, b, \alpha) = \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j d_i d_j \mathbf{x}_i^T \mathbf{x}_j, \quad (2.59)$$

subject to the constraints

$$\begin{aligned} (1) \sum_{i=1}^N \alpha_i d_i &= 0, \\ (2) \alpha_i &\geq 0 \text{ for } i = 1, 2, \dots, N. \end{aligned} \quad (2.60)$$

This way, the function to be maximized $Q(\alpha)$ depends only on the input patterns. After find the optimum Lagrange multiplies α_i^o , the optimum weight vector is obtained by

$$\mathbf{w}_o = \sum_{i=1}^{N_s} \alpha_i^o d_i \mathbf{x}_i, \quad (2.61)$$

where N_s is the number of SV and the SV corresponds to any pattern $(\mathbf{x}_i, d_i)$ in the training set that the Lagrange multiplier is nonzero. The bias can be calculated by

$$b_o = 1 - \mathbf{w}_o^T \mathbf{x}^{(s)} \quad (2.62)$$

which $d^{(s)} = 1$ and $\mathbf{x}^{(s)}$ is any support vector.

Thus, the discriminant function can be defined as:

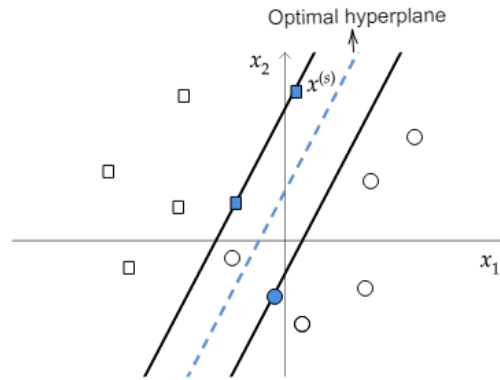
$$g(\mathbf{x}) = \text{sign}\left(\sum_{i=1}^N \alpha_i^o d_i \mathbf{x}_i^T \mathbf{x} + b_o\right). \quad (2.63)$$

Until now, it was considered that the problem was linearly separable, such way a hyper-plane would separate the two classes, without any errors. However, for real problems, this condition is often not respected. The margin flexibility, known as soft margin hyper-plane allows a training data that cannot be separated without errors, be divided by a hyper-plane with a minimal number of errors, as exemplified in Figure 8. This flexibility can be described by

$$d_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - \xi_i, \quad i = 1, 2, \dots, N, \quad (2.64)$$

which ξ_i are the slack variables. for $0 < \xi_i \leq 1$, the data point falls inside the region of separation, but on correct side of the decision surface and for $\xi_i > 1$, it falls on the wrong side of the separating hyperplane.

Figure 8 – Illustration of a soft margin hyperplane. One circle data point falls on the wrong side of the decision surface



Source: Adapted from Haykin (2008)

With this modification, the SVM soft-margin primal problem is defined as

$$\Phi(\mathbf{w}, \xi) = \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum_{i=1}^N \xi_i, \quad (2.65)$$

where C is the reciprocal of the regularization parameter, that controls the trade-off between complexity of the machine and the number of non separable points. When C is large, the designer of the SVM has a high confidence in the quality of the training set, when C is small, it is considered noisy and less emphasis should be placed on it. C is a SVM hyperparameter (HAYKIN, 2008).

Thus, the SVM soft-margin algorithm try to maximize the margin between classes and also try to minimize the slack variables ξ_i . Similarly to the hard-margin classifier,

soft-margin problem can also be solved using the method of Lagrange Multipliers. The dual optimization problem for soft-margin can be formulated as follows:

$$\begin{aligned}
 Q(\alpha) &= \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j d_i d_j \mathbf{x}_i^T \mathbf{x}_j, \\
 &\text{subject to the constraints} \\
 &\sum_{i=1}^N \alpha_i d_i = 0, \\
 &0 \leq \alpha_i \leq C \text{ for } i = 1, 2, \dots, N.
 \end{aligned} \tag{2.66}$$

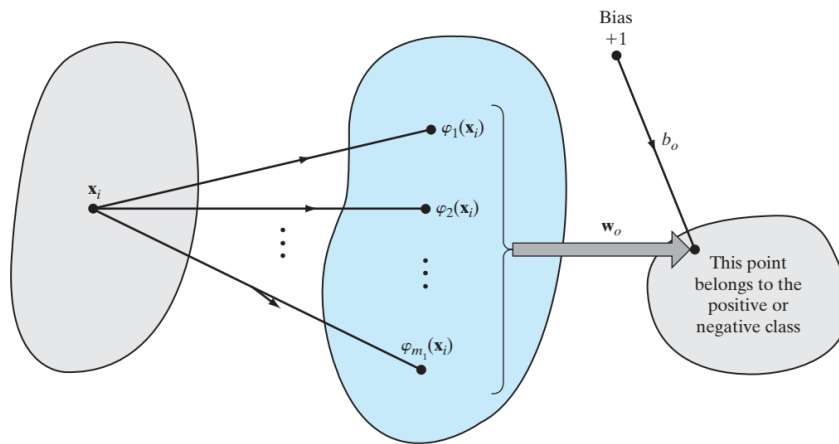
The only difference between this dual problem and 2.59 is the restriction to the Lagrange multipliers α_i . Except that, the computations of $\mathbf{w}$ and b is the same than the linearly separable case.

2.4.7.1 Inner-product kernel

After the previous explanation about the support vector machines, the description of SVM for pattern recognition tasks is described in Figure 9 and can be summarized in two main operations:

1. Nonlinear mapping of an input vector to a high-dimensional feature space;
2. Construction of an optimal hyperplane for separating the categories described by the features discovered in step 1.

Figure 9 – Mappings in a support vector machine for pattern classification



Source: Haykin (2008)

Considering a continuous and symmetrical function $K(x, y)$, that computes the inner product of the images produced in the feature space under the embedding ϕ of two

data points in the input space

$$K(x, y) = \phi^T(\mathbf{x}_i)\phi(\mathbf{x}), \quad (2.67)$$

which $\phi^T(\mathbf{x}_i)\phi(\mathbf{x})$ is the inner product of the image of feature vector of the input pattern and the support vectors, respectively.

The discriminant function can be rewritten as

$$g(\mathbf{x}) = \text{sign}\left(\sum_{i=1}^{N_s} \alpha_i^o d_i K(\mathbf{x}_i, \mathbf{x}) + b_o\right). \quad (2.68)$$

In this dissertation, the kernel used is the Gaussian function

$$K(x, x_i) = \exp\left(-\frac{1}{2\sigma^2} \|\mathbf{x} - \mathbf{x}_i\|^2\right). \quad (2.69)$$

2.4.7.2 SVM for multi-class problems

In previous sections, the SVM has been defined for binary classification problems. However, real problems often require discrimination for more than two categories. Multi-class problems can be decomposed into a series of binary problems and the standard SVM can be directly applied. Two simple and common approaches are the one-versus-rest (OVR) and one-versus-one (OVO). There are other more complex approaches that addresses the drawbacks of those classical approaches and even models that combine multiple binary optimization problems in one single objective function, however larger computational complexity is required (WANG; XUE, 2014). In this dissertation, the OVO approach is used.

OVR approach designs k separate binary classifiers for k -class classification. The m_th binary classifier is trained using the data of the m_th class as positive labeled examples and the remaining $k-1$ classes as negative labeled. During test, the class label is determined by the classifier with maximum output value. The main disadvantage is the imbalanced training set for each binary classifier, losing the symmetry of the original training set.

Considering a balanced multi-class dataset, the training patterns positive to negative ratio for each individual classifier in OVR approach is $\frac{1}{k-1}$. This way, the greater the number of classes, the greater is the imbalance between positive and negative classes. Thus, OVR should not be recommended for a problem with a large number of classes. The OVO approach solves this imbalance problem, because one classifier is trained for each pair of classes. However, for this, $\frac{k(k-1)}{2}$ individual classifiers are necessary. During prediction, all the classifiers are applied to the input pattern and one vote is accounted for each classifier winner class. The most voted class is assigned to the pattern.

2.4.8 Optimum Path Forest

The OPF classifier was proposed by Papa et al. (2007). This classifier is natively multi-class and capable of handling some degree of overlapping between classes. The training set is modeled as a graph, whose nodes are the samples and the arcs are defined by some adjacency relation and weighted by the distance between the feature vectors. The idea is to identify prototypes in each class such that every sample is assigned to the class of its most strongly connected prototype.

Based on the training set, The OPF classifier is designed creating a optimal partition of the feature space such that unseen patterns (testing samples) can be classified according to this partition. It is computed by the image foresting transform (IFT) (FALCAO; STOLFI; LOTUFO, 2004).

Consider the training set $Z = (\mathbf{x}_i, d_i)_{i=1}^N$, which $\mathbf{x}_i$ is a vector in the feature space and d_i is class label. In the first step of OPF training, a complete undirected graph $A = (Z \times Z)$ is built. In this graph $\mathbf{x}_i$ represents the nodes and the edge weights are the distances between the training patterns, using a predefined metric, like Euclidean distance. The next step is to find the minimum spanning tree (MST) of A using any algorithm with this purpose, such as Kruskal's and Prim's algorithm. The M nodes of the MST that are connected to different classes nodes are defined as the prototypes and together they compose the Z_p set. The $L = N - M$ non-prototypes nodes form the non-prototypes set Z_{np} (SOUZA; LOTUFO; RITTNER, 2012).

A path $\pi_{s,t} = \langle s, a_i, a_{i+1}, \dots, t \rangle$ between nodes s and t is defined as a sequence of connected nodes between s and t . The next step in the fitting process is to calculate the path cost C of every training pattern to every prototype and assign each training pattern to a tree rooted in the prototype that offered the minimum cost, defined as

$$C(\mathbf{x}_l) = \min\{f_{cost}(\pi_{\mathbf{x}_l, \mathbf{x}_{pi}})\}, \forall \mathbf{x}_{pi} \in Z_p, \quad (2.70)$$

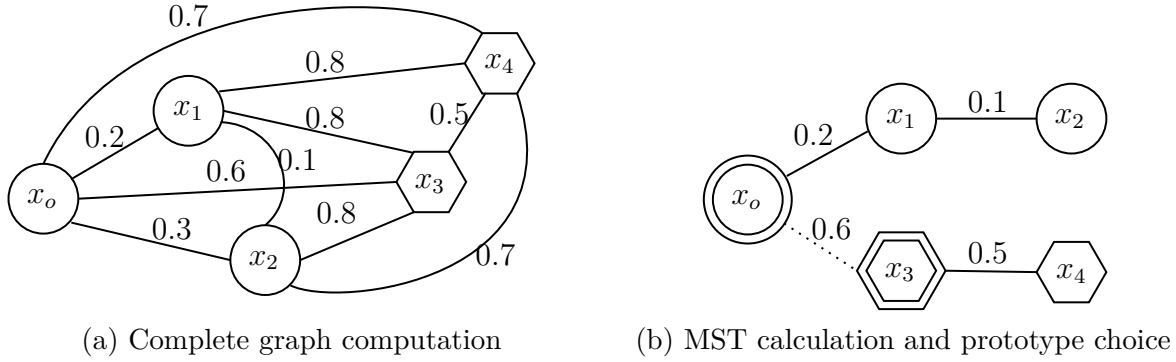
with

$$f_{cost}(\pi_{s,t}) = \max\{d_j\}, \forall j \in \pi_{s,t}. \quad (2.71)$$

The combination of Equations 2.70 and 2.71 is the usual OPF classification problem formulation and can be solved using Maximum Dijkstra's algorithm adapted to multi-source case (FALCAO; STOLFI; LOTUFO, 2004), where the sources are the prototypes. In case of ties, the training sample is assigned to the tree with lower number of edges in the path.

The OPF fitting is illustrated on Figure 10. There are five training patterns, x_o, x_1 and x_3 from class circle and x_3 and x_4 from class hexagon. In Figure 10a there is the complete graph of the first stage of the fitting algorithm and in Figure 10b the MST is displayed with the prototypes highlighted by double contour line.

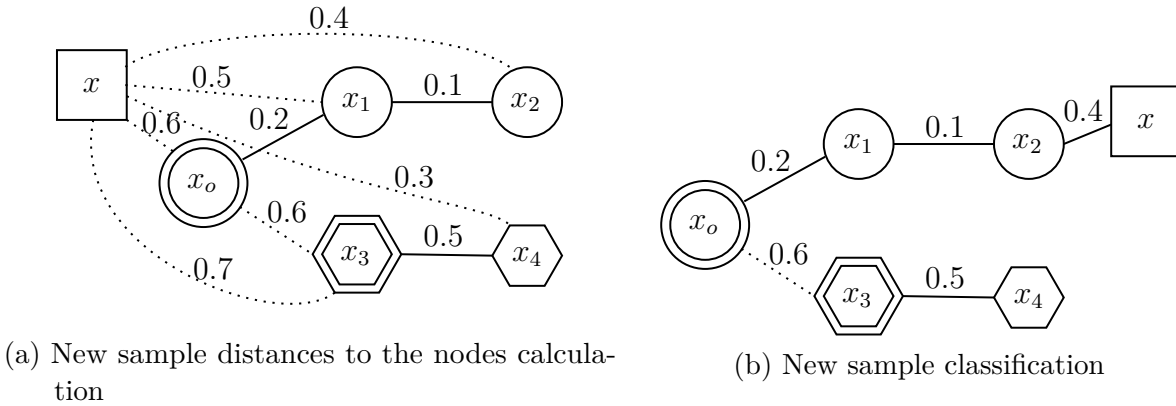
Figure 10 – OPF fit illustration



Source: Adapted from Souza, Lotufo and Rittner (2012)

The OPF prediction consists of calculating the distance from the new pattern x to all the tree nodes and assign the label of the closest prototype to x . This operation is exemplified on Figure 11. Considering the OPF fitting example illustrated in Figure 10. After the new pattern nodes distances are calculated in Figure 11a, x is assigned to class circle in Figure 11b, because it is closer to x_o than to x_3 .

Figure 11 – OPF prediction illustration



Source: Adapted from Souza, Lotufo and Rittner (2012)

There are similarities between OPF and KNN, when you consider they both use distance metrics and are non parametric. Souza, Lotufo and Rittner (2012) showed 1-NN and OPF are theoretically equivalent when all training patterns are used as prototypes and the simulation results confirmed this hypothesis, despite OPF boundaries are smoother. However, for the example described in Figure 11, if 1NN was used, x would be classified as hexagon, because its nearest neighbor would be x_4 . Thus, this example illustrates the differences in classification results between KNN and OPF.

2.5 Embedded systems

An embedded system, can be defined, in a very simplistic way, as any device that includes a programmable device not intended to be used as a general-purpose computer (WOLF, 2008). Marwedel (2006) defines embedded systems as information processing systems embedded into a larger product and usually not directly accessible to the user. Washing machine, cell-phones, smart TVs and automobiles are examples of embedded systems applications. Since the earliest days of computing, computers have been embedded into applications. In the beginning, they were physically extremely large compared to today's computers, but their usability in replacing human control and analysis remains.

A microprocessor is a single-chip CPU, that can be programmed to implement a required functionality and then be reprogrammed for use in other applications as well. Application Specific Integrated Circuit (ASIC) design is expensive and time consuming, and for this reason microprocessors become widely applied in the development of digital systems. Furthermore, it allows product upgrades and the development of families of products with different functionality, without changing the hardware. They are available in very different configurations, that should fit the application to optimize performance and cost. Even when hardware must be modified for products next-generation, it may be possible to reuse software or parts of it, reducing development time and cost (WOLF, 2008).

Depending on the application, embedded computing systems might execute complex algorithms or handle user interfaces including multiple menus. Other requirements can be real time execution, what means the system should be able to meet strict deadlines, or present a multi-rate behavior, to handle operations that happen at very different rates, but should be synchronized at some point. However, the challenge really starts when considering costs constraints that hugely impacts on type of microprocessor, amount of memory, peripherals devices or auxiliary units. Another important constraint is power consumption, which affects the cost, when considering power supply, autonomy, for battery powered devices and also heat dissipation.

In embedded systems design, obviously, there is not one device that fits all kinds of applications. Different requirements lead to different trade-offs between performance and power, hardware and software. A common form of classifying microcontrollers is based on their word size. An 8-bit microcontroller is usually applied in low-cost applications that handles I/O devices and do not need intensive computation. 16 bit devices are often used for more sophisticated applications, that require longer word lengths and 32 bit microprocessors offer very high performance for computational intensive applications (WOLF, 2008).

The study of embedded system design includes three complementary aspects:

architectures, applications and methodologies. Because it covers hardware and software combined, the knowledge about hardware architectural aspects, like special-purpose units and multiprocessors, is as important as software architectural ones, that determine, for instance, how to take advantage of parallelism to improve performance and cost. Understanding the application details and context is critical to determine the requirements and specify the hardware and software architecture. The interconnection between those aspects is fundamental to take advantage of the system at maximum, optimizing the design. By last, well defined methodologies guide decision making during development, contributing to save time and make the design process more repeatable (WOLF, 2014).

After application specifications well defined, the hardware should be chosen. A very important component is the processing unit, that can be a processor, an ASIC, configurable logic like FPGAS, among others. As already explained, processors offer the largest amount of flexibility. They can be optimized for the application, like digital signal processors (DSPs), for intense data processing and tight deadlines cases. For those devices, power-efficiency approaches configurable logic values (MARWEDEL, 2006). Digital signal processing in embedded systems is introduced in the following subsection.

2.5.1 Embedded digital signal processing

DSPs have their architecture optimized to perform high-speed numeric calculations like vector addition, subtraction, multiplication and division. One of the most common operation required in digital signal processing applications is array multiplication, which is used during convolution and correlation. To speed up, a dedicated Multiply accumulate (MAC) unit may be implemented in hardware. This way, a MAC instruction in a DSP can take only one instruction cycle (VENKATARAMANI; BHASKAR, 2011). Another instruction very important to DSP is SIMD (Single Instruction Multiple Data), which perform multiple identical operations in a single cycle instruction (STMICROELECTRONICS, 2018).

Digital signal processing can be separated in two categories: fixed point and floating point. Those concepts are related to the manner the numbers are presented. In the first case, with a fixed number of digits after and before the decimal point. Meanwhile with floating point, the decimal point can "float" relative to the significant digits of the number. The exponentiation assures a much larger dynamic range that is important when processing large data sets or when the range may be unpredictable. Another advantage of floating point is related to precision. The gap between any two numbers is approximately ten million times smaller than the value of the numbers (ANSI/IEEE Std. 754 standard format), with larger gaps for larger numbers and smaller gaps between small numbers. Furthermore, fixed point development is usually more difficult and slower, it is necessary more workaround to avoid overflow and compensate the limited number range, what can

increase the time-to-market. However, it is usually faster and also more power efficient (Analog Devices, 2018).

Floating-point calculations require a lot of resources to be performed. If all these operations are done through software, many instruction cycles are necessary. To perform floating-point operations efficiently, for applications like control systems and digital signal analysis, a FPU is necessary to execute arithmetical operations entirely by hardware, most of them in a single cycle (STMICROELECTRONICS, 2016).

2.5.1.1 DWT on embedded systems

In literature, there are not many examples of DWT implementations on embedded systems, being the majority made in general purpose computers. Ukil (UKIL; BARLOCHER, 2007) describes a fixed point implementation of the DWT using TMS320VC5510, a 16-bit Texas Instruments DSP. In this paper, the performance of two DWT algorithms are compared: the classical technique, using convolution and decimation, and the other using lifting operations.

The main characteristic of the lifting algorithm consists in decomposing the low-pass and high-pass filters into a sequence of superior and inferior triangular matrices. The advantages of this algorithm are the in-place execution of the DWT and the symmetric inverse and direct Wavelet Transform (ANDRA; CHAKRABARTI; ACHARYA, 2002). However, the performance comparison (UKIL; BARLOCHER, 2007) showed that the implementation based on convolution is about 3 times faster than the lifting scheme and it also occupies less program memory. Additionally, one can not see the DWT parameters used to generate the execution time information, like the number of levels in the decomposition, the mother-wavelet or the input data length. This way, it is hard to scale the results to other applications.

Related to the field of induction motor failure detection, Ordaz-Moreno et al. (2008) developed an online system to detect broken bars using DWT implemented to an FPGA. The selected platform is based on an XC3S200 Spartan-3 FPGA from Xilinx. The input data has 8192 samples, 6 levels of decomposition is performed and it takes only $16.66\mu s$ due FPGA parallel processing capacity.

2.5.2 Embedded machine learning

In the last years, due MCU prices drop, rapidly-increasing capability of computers, greater data capturing ability, often in form of inexpensive sensors already embedded in many products, the development of new algorithms that improve ML applications accuracy and the creation of toolkits that makes the ML design easier culminated in the increase of sensor-driven embedded artificial intelligence in products (GIGAOM, 2019).

The automotive field is leading this trend. The average non-autonomous vehicle has now 100 sensors, sending data to 30 - 50 microcontrollers. Luxury car may have twice as many, and autonomous vehicles increase the sensor count dramatically. Industrial equipment are also becoming "smart" including functionality like condition monitoring and predictive maintenance. Other products like fitness monitoring and vacuum cleaners are also following this trend. What all those applications have in common is the need to use real-time, streaming and complex sensor data to find the signature of specific events and conditions, locally, in embedded systems, usually keeping the prices low (Reality AI, 2018).

Haas and Davies (2019) white paper presented an study that suggests in 2017, 324 million embedded systems devices used some form of AI. Two thirds were edge-compute systems such as Internet of Things Applications and one third were mobile devices. Besides the advantages of running the ML in edge compared to cloud based approaches, described in Chapter 1, perform ML in the edge is not a choice, but a need. With the increase in connected devices, and the fact that ML tasks are usually data intensive, there is not enough bandwidth in the world to support sending all the data to server farms for processing. Furthermore, many applications, such as autonomous vehicles, cannot deal with the traditional cloud computing model latency.

In this scenario there are some challenges: Real world data is noisy and full of variation, fast response might be hard to achieve, when the application requires complex algorithms and the available resources are limited by physical (size and weight), power and cost constraints.

Recently, deep learning has gotten attention due its stunning results in many applications. But is also has drawbacks like very large datasets for training and specialized hardware. For this reason, depending on the application, other approaches can be more appropriate considering cost, size and power constraints.

Another important aspect that differentiates the development of machine learning solutions in general and embedded ML applications are the compromises and tradeoffs. The data acquisition sampling rate and number of points captured impacts significantly the memory and clock-speed of the target platform. Since these also affect the model accuracy, there is a trade off between accuracy, computational requirements and response time that should be evaluated.

An ARM commissioned research survey (HAAS; DAVIES, 2019) about the main hardware used for embedded AI/ML applications in different sectors, including IoT, industrial, automotive and mobile computing pointed CPUs (40 %) as the main option for development of the applications, followed by GPU, FPGA, Dedicated Machine Learning Processors and DSPs. The main reasons for CPU leadership is their flexibility, security and cost efficiency. According to Haas and Davies (2019), the perspective is that CPU

and MCU will remain preeminent, while hybrid systems where the CPU takes advantage of co-processors, like NPU and GPU for specific tasks are emerging.

2.5.3 Embedded induction motor failure detection

As already stated, the induction motor failure detection research field is very diverse in terms of techniques and algorithms proposed. However most of the studies stops in the proposition of the algorithm and there are significant less examples of those algorithms executed in embedded systems, their performance and cost in constrained devices.

Most of examples found in literature uses FPGAs (RODRIGUEZ-DONATE et al., 2008; PANIGRAHY et al., 2016; KARIM; MEMON; HUSSAIN, 2018), to take advantage of the parallel processing capacity. There are also examples using microcontrollers, like Guven and Atis (2018), that uses ARM Cortex-M3 to detect broken bars using current signal and Grey Relational Analysis algorithm. In another paper (CHANG YU-MING JHENG; HSUEH,), a Cortex-M4 is used to implement an online motor condition monitoring using voltage, current and vibration signals .

Considering the short-circuit defect, Vieira et al. (2016) presented an embedded solution that detect the failure with 83 % accuracy using current signal, FFT and MLP. The platform used is the dsPIC 33EP512MU810. It uses 41040 bytes of RAM and 19365 bytes of FLASH. The data is acquired at 1 kHz sampling rate for 4.096 s and the processing takes 32 ms.

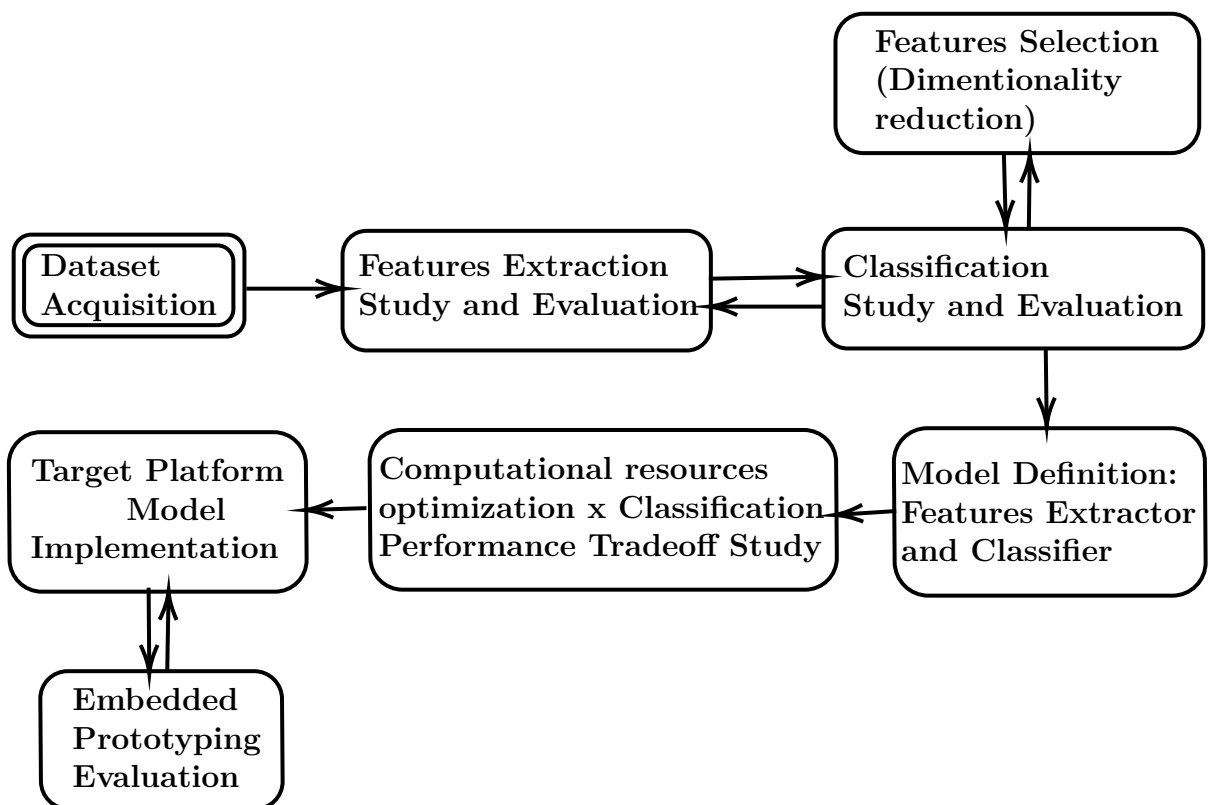
3 Design methodology and data acquisition

In this chapter, the methodology applied to the development of an embedded prototyping for induction motor short circuit failure detection, using the leakage flux as the input signal, is described.

3.1 General view

In general, for applications that combines signal processing, machine learning and embedded systems as target platforms, it is possible to establish a sequence of steps to develop a solution. Despite many particularities relative to the application, like the signals available, types of sensors used, acquisition time and frequency, there are some common characteristics such as: the need to extract features to describe the event, state or relation that the system should be able to model; to define the classifier that best fits the application, considering classification performance and cost; to define the platform, among others.

Figure 12 – Machine Learning Embedded Application Design



The flow chart of Figure 12 illustrates the investigation and development process.

It consists of the following steps:

1. **Data acquisition** to compose a dataset to be used for training and validation of the model. It is fundamental to ensure the data quality, minimizing noise interference and aliasing errors with analog filtering prior the data capturing. It is also important the dataset represents as much as possible the diverse situations that can happen in real application to ensure the model is robust and generic enough to present good results for new data;
2. Study about the application to **identify characteristics that describe well the problem**. In this stage, deep knowledge of the application can speed up the process, helping to choose the **features extraction** or domain conversion techniques that might be adequate for the application. But if it is not the case, there are techniques that help to understand the dataset, like pairwise plots, histograms, statistical analysis, among others. This stage is similar to any other machine learning problem. The difference is that the aspects that could impair or increase the cost of a real embedded application should be taken into account;
3. **Evaluation of different combinations of feature extraction and machine learning techniques** and the model performance. Improvement in feature extraction or **selection** and machine learning tuning techniques should be analyzed until satisfactory results are achieved. This and the previous stage is done in workstation, but it is important to be aware of the target platform constraints.
4. **Computational resources usage analysis and optimization exploration**, considering the trade-off between performance and the project requirements, like response time, cost, power consumption, hardware specification, among others. The model is defined when an acceptable compromise between the different requisites is found.
5. **Target platform model implementation**. In this stage, the algorithm developed in the workstation is replicated in the target platform. Depending on the signal processing, the classification techniques and target device capabilities, it can be very challenging. The use of optimized libraries or specialized hardware might facilitate and give better results than implement everything from scratch. The cost, flexibility, development time and performance requirements for the different possibilities should be analyzed;
6. **Embedded Prototyping Evaluation**. In this stage the results obtained in the target platform are compared to the workstation to ensure they are equal or the deviation caused by variable precision differences, data type or other adjustment

applied during the previous stage is acceptable. The metrics related to the project requirements are evaluated and improvements for next versions are also investigated.

Following a project methodology and taking into account some aspects of the embedded systems applications should speed up the development of the prototype, reduce re-work in the embedding phase and save computational resources.

The methodology described in this section has also been applied to a different application: pollutant detection in energized polymeric insulators (CUNHA et al., 2018). It suggests the applicability of this methodology to diverse problems with the common characteristics previously described. More information about this application and the obtained results are available in Appendix A.

In the next section, the data acquisition, represented as the first step of the Figure's 12, is described.

3.2 Data acquisition

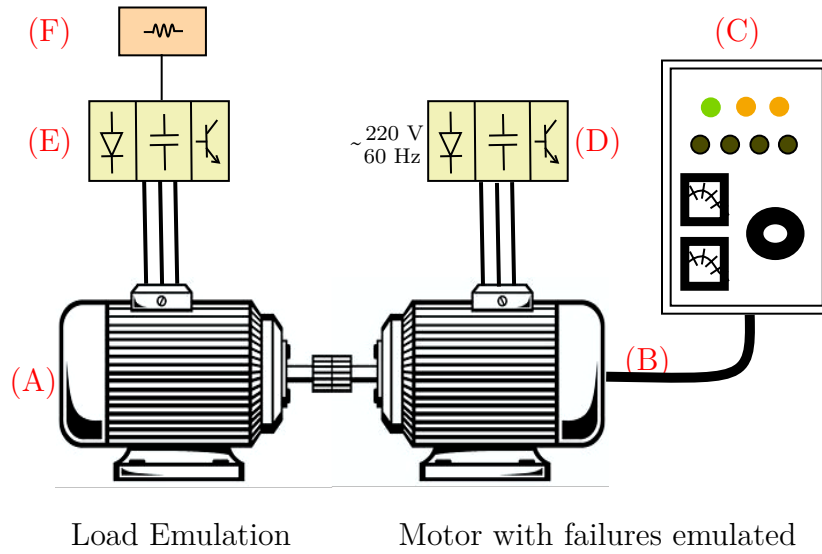
In this section, it is described the first stages in the development of a model to detect stator winding short-circuit failure in induction motors. It includes the emulation of the failure and operating conditions using a test bench and the signal acquisition to compose the dataset.

3.2.1 The test bench

The test bench is composed of two similar triphasic squirrel cage induction machines, two frequency converters, a failure emulation control panel and a resistor load bench. In Figure 13, there is a diagram of the test bench in operation. The left machine (A) emulates the mechanical load and operates as a generator, power supplying the resistor load bench (F). The machine on the right operates as a motor (B) and it was properly prepared to enable the emulation of stator winding inter-turns short-circuits. The frequency converters are used to drive the induction machines (D and E). This way, the machines can work at different driving frequencies. The failure emulation control panel (C) emulates connections for the short-circuit between stator coil turns. It also includes three current sensors and two analog current meters to allow the visual monitoring of the phase currents and the short-circuit current. In Figure 14, there is a picture of the set.

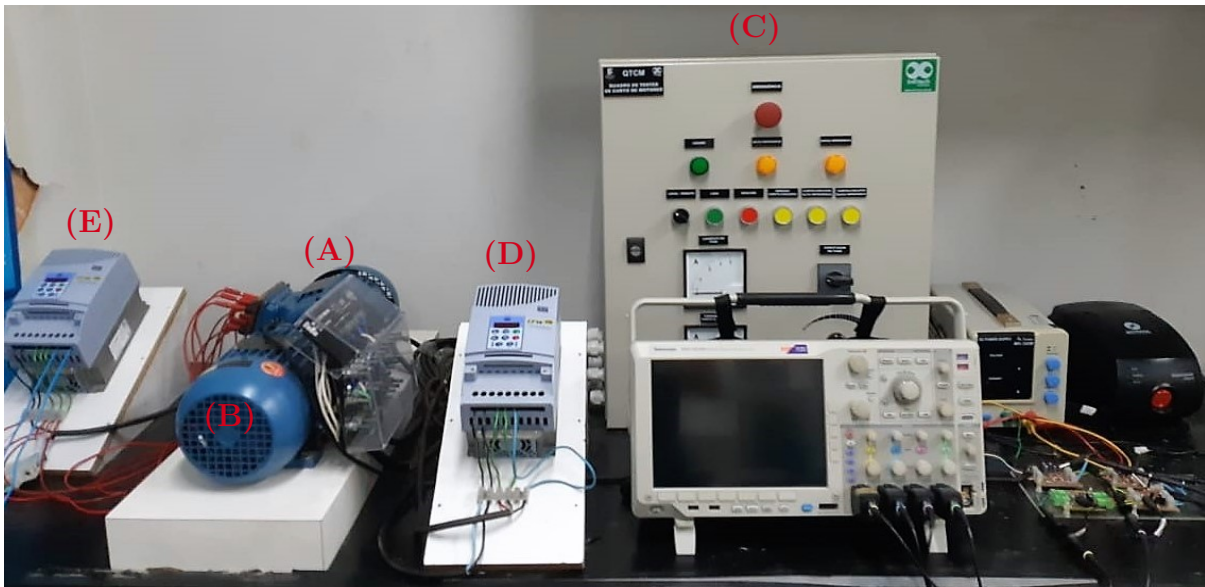
The induction machines used have the following specifications: 4 poles, 1 HP of mechanical power, delta configuration, 220 V of supply voltage and 3 A of rated current. The frequency converters are both WEG CFW-08 (WEG, 2019), with 2.5 kHz PWM switching frequency.

Figure 13 – Test bench diagram



Source: Author

Figure 14 – Test bench photography



Source: Author

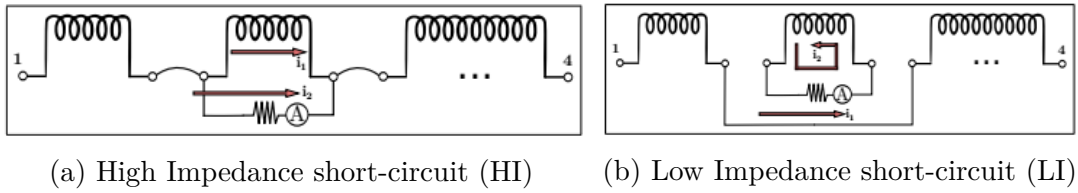
3.2.1.1 Failure emulation

The electric machine used was prepared by a specialized company. Its stator circuit was re-winded, making possible to access the derivations of the winding, in order to insert inter-turn short-circuits. Different levels of short-circuit can be emulated, from very incipient defect to the more severe situation. The same motor was used by Oliveira, Silva and Medeiros (2013) and the complete test bench by Rebouças Filho et al. (2018), to

investigate the detection of the same type of failures in induction generators.

Two types of failures are emulated: High Impedance (HI) and Low Impedance (LI) short-circuit. The first one represents the initial stage of the failure, in which the electric insulator is beginning to degrade and a parallel current path arises. It causes overheating in this region until the complete insulator degradation. The second type represents a full short-circuit. The current flows in the new path and a voltage is induced in the shorted coil. Of all the types of failures emulated, the short-circuit current intensity is limited to its rated value, using a variable resistor, to prevent permanent damage to the winding, as would happen in a real serious short-circuit. Thus, the most severe type of low impedance condition emulated is, actually, a more intense high impedance failure. However, it is expected that if the model is able to identify a softer low impedance short-circuit, it can detect a similar one of higher intensity. The two types of emulated short-circuits are represented in Figure 15.

Figure 15 – Representation of emulated fault types



Source: Adapted of Nascimento (2018)

Different intensity levels are also emulated depending on the total number of shorted coil turns. Three levels are considered: 1.41 %, 4.81 % and 9.26 % of the stator winding. Combining the type and intensity of the defects, there are, respectively, HI-1, HI-2 and HI-3, for high impedance, and LI-1, LI-2 and LI-3, for low impedance failure.

The complete dataset is composed of 7 classes (motor in normal condition and 6 failure conditions). To increase the representation of real situations and improve the model robustness and generalization, different combinations of driving frequency and load levels are applied.

3.2.1.2 Load emulation

The model should be able to detect the failure under different load conditions. Therefore, three load conditions are applied to compose the dataset: 0 %, 50 % and 100 %. To emulate the load, an electrodynamicometer is coupled to the motor. It is another induction machine with the same specifications of the machine under test, directly coupled to the motor shaft, as shown in Figure 16a.

This second machine operates as a generator. The mechanical load imposed on the motor is proportional to the power requested to the generator. A resistor load bench

is the load of the electrical generator (Figure 16b). The frequency converter is used to drive to the generator a rotating field speed lower than the rotor speed, forced by the motor, resulting in negative slip. Another function of the frequency converter that drives the generator is to provide reactive power to produce the rotating magnetic field and start the power generation.

Figure 16 – Equipment used to emulate mechanical load

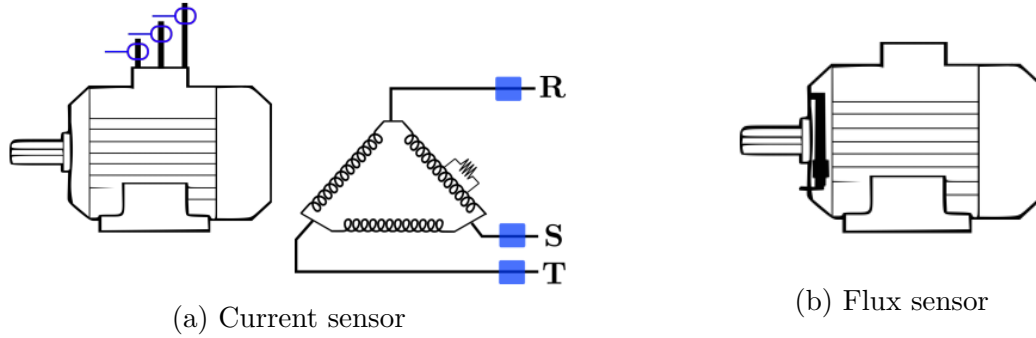


3.2.2 Dataset composition

To monitor the axial leakage flux, a coil of 100 turns of copper wire 24 AWG was placed around the motor shaft. The electromotive force generated, as described by Lens Law is collected. The same flux probe was used by Sousa and Medeiros (2017) to detect broken rotor bars and by Sousa et al. (2018) to monitor short-circuit failures in induction generators.

The currents of the 3 motor phases were acquired using current transformers (CT) model SCT013-030. They present 3 % non linearity, which ensures a measurement reliability between 10 % to 20 % of the sensor nominal current (0 to 20 A). The output is an alternated signal with amplitude range of 0 V to 1 V (YHDC, 2010). In Figure 17, there are 2 diagrams illustrating the position of the current (17a) and axial flux (17b) sensors on the motor.

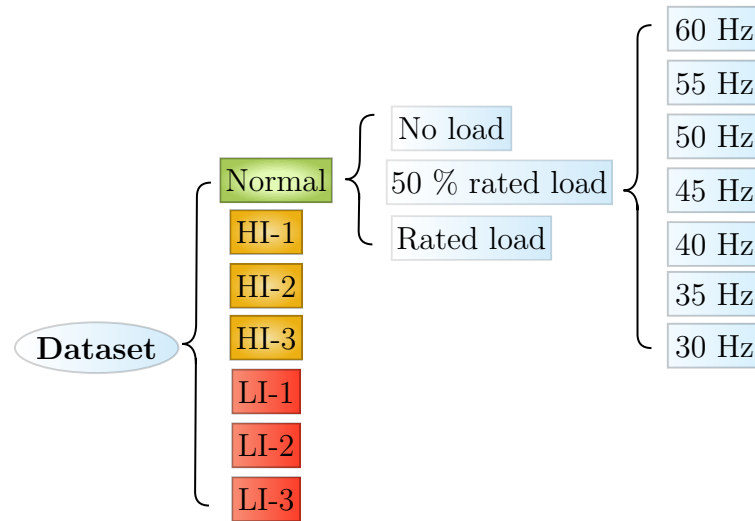
Figure 17 – Sensor placement diagrams



Source: Nascimento (2018)

The signals from flux and current transducer are filtered, conditioned and digitalized, meanwhile the motor is operating under a specific configuration of frequency, load and failure (type and intensity). A total of 2590 patterns were acquired: 350 of normal class and 2240 of failure conditions. In Figure 18 the variations in failure type, load and frequency are illustrated for normal class. The same goes for the other classes.

Figure 18 – Distribution of the patterns in the dataset



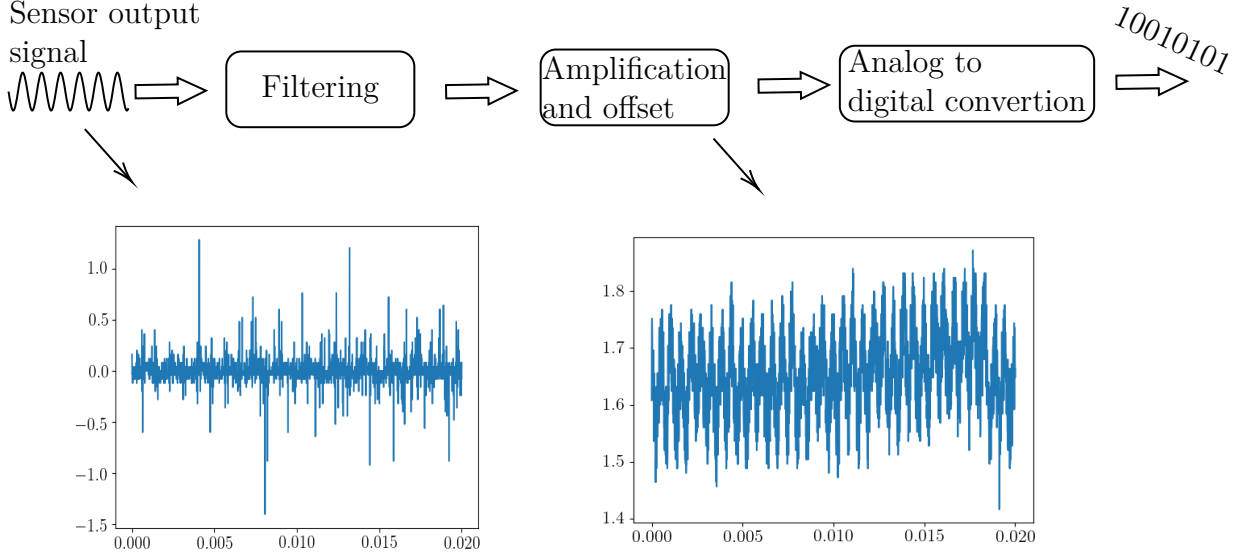
Source: Author

3.2.2.1 Signal filtering and conditioning

The diagram in Figure 19 describes the analog processing of the signals from sensor output to analog to digital converter input. Before transducers signals digitalization, it is important to filter the signal to limit its frequency range, to prevent aliasing distortion (TAN, 2008) and also minimize the interference of noise originated from the frequency converters or other sources. Thus, a second-order low pass Butterworth filter, using a Sallen-Key configuration was applied. A cutoff frequency (f_c) of 1 kHz was chosen due

the prior knowledge the frequency band of the signal. The Butterworth filter was selected because it gives the best compromise between attenuation and phase response (Analog Devices, 2007) and the flattest frequency response in the pass-band.

Figure 19 – Signal conditioning before analog to digital conversion



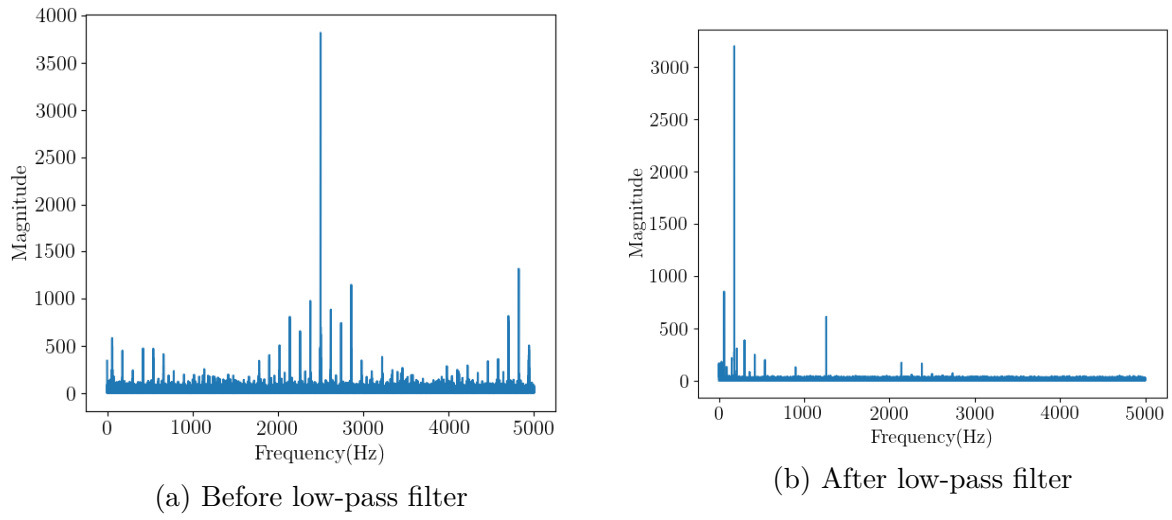
After the signals are filtered, they are amplified to bring their amplitude to optimal levels to drive the ADC. To compose the database used in the development of the model to detect short-circuit failures using the axial flux data, the signals were acquired using an oscilloscope. However, as the developed model is intended to become an embedded application, the signal conditioning phase was designed considering the requirements of the ADC of common microcontrollers. That is 0 V to 3.3 V ADC input range. This way, the filtered signals have their DC levels shifted to 1.65 V, and then amplified.

An example of the flux signal before and after signal conditioning stages is plotted in Figure 19. The difference between the signals is very significant even in time domain. Before filtering, the signal is mixed with high frequency noise and it is hard to distinguish the flux itself. Figure 20 presents the frequency spectrum of the same signals. Comparing them, it is easy to verify that the components originated mostly by the frequency converters used to drive the machines, about 2.5 kHz and its harmonics are suppressed or significantly attenuated after filtering.

3.2.2.2 Digitalization

The analog to digital conversion was performed using the Tektronix MSO4034B oscilloscope, with 10 kHz sampling frequency and 10 s total acquisition time, which results in 100000 samples per pattern. The resolution of the analog to digital converter of this equipment is 8 bits (TEKTRONIX, 2016).

Figure 20 – Frequency spectrum of flux signal before and after analog low pass filtering



3.3 Conclusion

In this section has been presented a general view of the methodology applied to the development of the present research. Also, the dataset acquisition process, considering analog and digital aspects are described in details.

In the next chapter, the detection of the short-circuit failures using pattern recognition with time domain features is investigated,

4 Time domain analysis

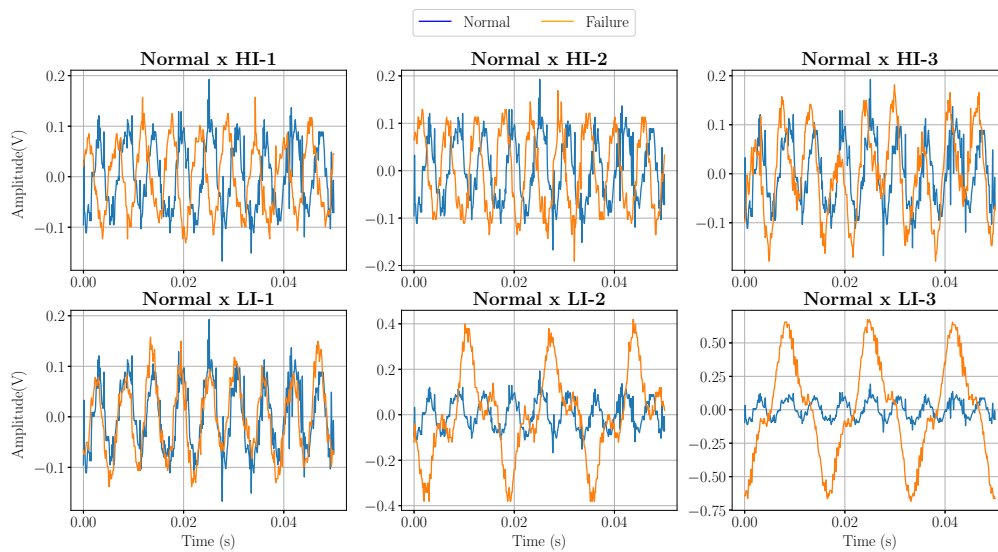
In this chapter, it is described the study about using statistical features extracted directly from the original time domain signal and the results obtained using different machine learning techniques, like Bayes classifier, multilayer perceptron and support vector machines. The effect of including information not extracted from the flux signal is also evaluated.

4.1 Short-circuit failure and leakage flux

The effects of the stator winding inter-turn short-circuit in the axial flux are analyzed based on the data acquired from the short-circuit emulation test-bench. In Figures 21, 22 and the flux of a normal machine is compared to the flux of each one of the failure types in two different conditions: 60 Hz without load and 60 Hz full load.

By visual inspection of the time domain signal, it is easy to notice the differences between normal and LI-2 and LI-3 signals. However, it is hard to differentiate the healthy signal from the rest of failures for motor operating at 60Hz without load. The same behavior occurs for the motor operating at 60 Hz full load.

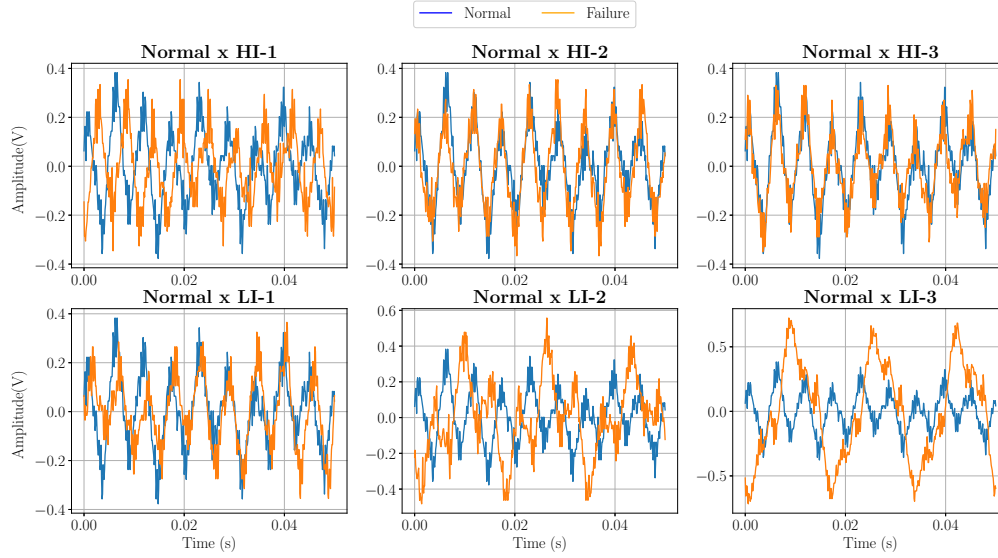
Figure 21 – Comparative of the axial flux for normal condition versus each type of failure. The motor operates without load, 60Hz



Source: Author

Statistical features extraction is one of the main time-domain signal analysis techniques (LEI, 2017). Skewness, kurtosis, mean absolute deviation and median absolute

Figure 22 – Comparative of the axial flux for normal condition versus each type of failure. The motor operates with full load, 60 Hz



Source: Author

deviation were selected to perform the initial investigation. In Figure 23 there are the pair plots of those features and the kernel density estimates of the feature values for each one of the classes. Analyzing the skewness and kurtosis combinations only, all the classes seem to be very overlapped. When mean absolute deviation and median absolute deviation are included, the LI-3 and LI-2 patterns start to concentrate in some areas. However, the overlap between normal and more incipient failures still significant. LI-3 seems to be well characterized by MeanAD and Median AD, meanwhile LI-2, by MeanAD and Kurt.

4.2 Pattern classification

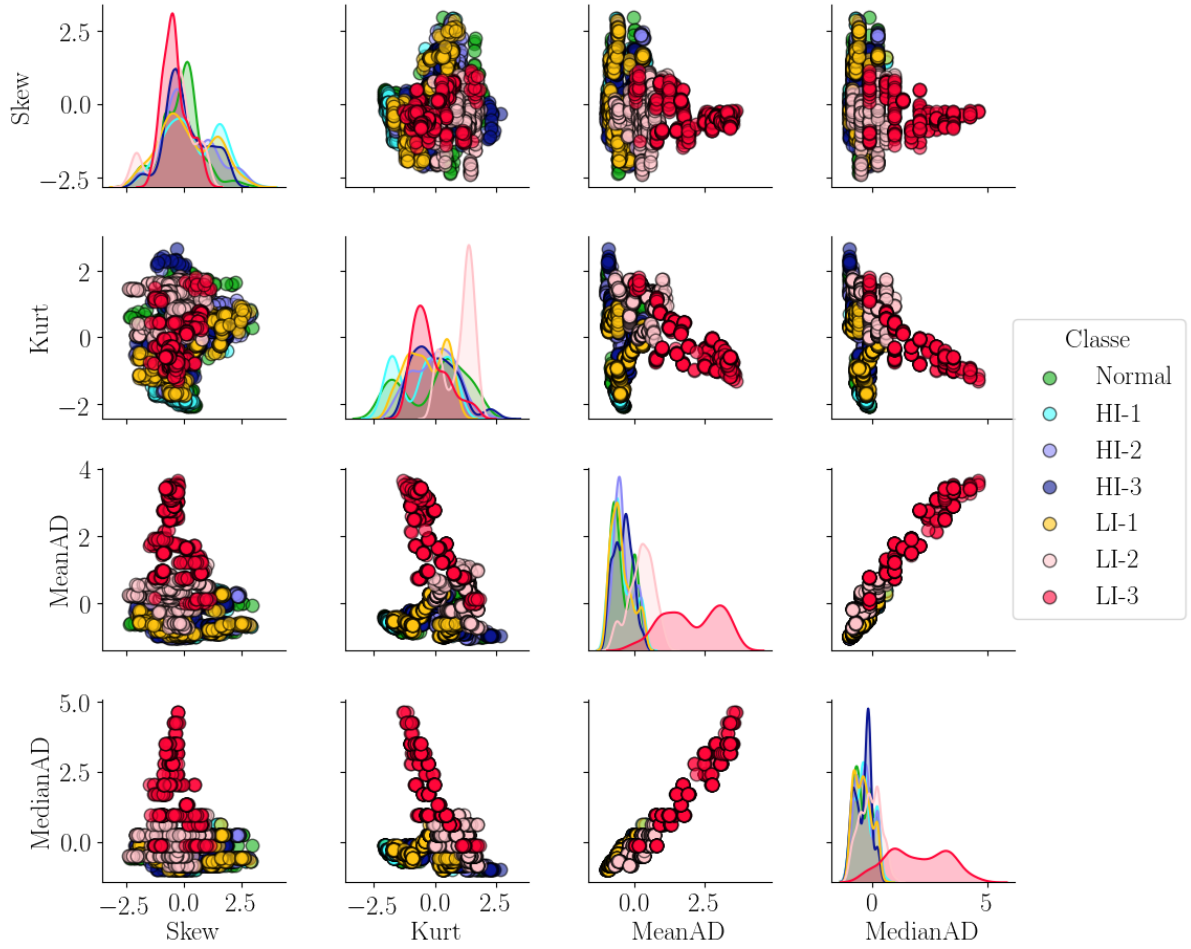
In this section, some classifiers were applied to the described dataset. To make a fair comparison, all the experiments follow some rules:

- Standard scaling was applied, that ensure unitary standard deviation and zero mean for each feature:

$$x_{nj} = \frac{x_{nj} - \mu_j}{\sigma_j}; \quad (4.1)$$

- 80 % of the data was used to train all the classifiers and 20 % to evaluate. The patterns of the two datasets are randomly chosen;
- To all the classification results presented, 20 independent executions of training and testing were performed;

Figure 23 – Pair plots of the four features: Skewness, kurtosis, mean absolute deviation and median absolute deviation



Source: Author

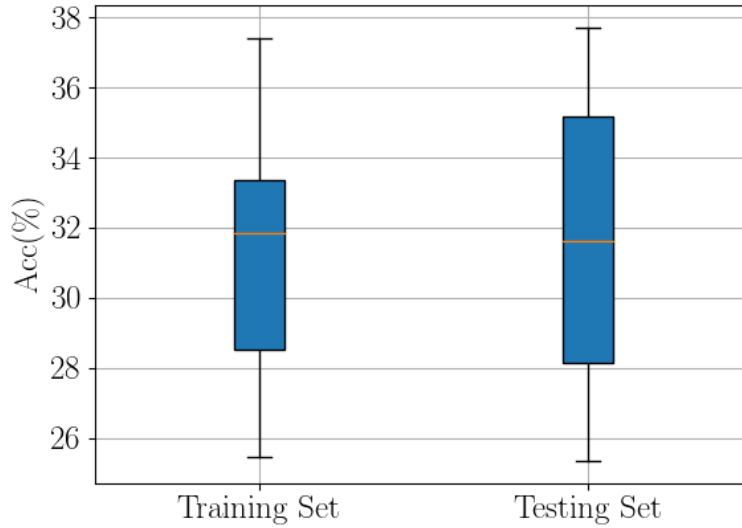
- For the parametric classifiers, the hiperparameters were chosen using a grid search with 10 - fold cross validation. The grid search was repeated 20 times and the most voted configuration is chosen.

4.2.1 Perceptron

Despite the visual inspection of the dataset suggests this problem is not linearly separable, the Perceptron was applied to verify it. To choose the hyper-parameters learning rate (η) and number of epochs (N), a grid search with 10-fold cross validation was applied using $\eta = [0.5, 0.4, 0.3, 0.2, 0.1, 0.05, 0.03]$ and $N = [200, 400, 600, 800, 1000, 1200, 1400]$. The learning rate exponentially decays until 0.0001. The grid search was repeated 20 times, and the most voted configuration, $N=1200$ and $\eta = 0.05$ was chosen. The results for 20 executions of training/testing with the chosen parameters is displayed on the box - plot of

Figure 24. The accuracy achieved is very low, about 32 %, what confirms that the problem is not linearly separable and more powerful (non linear) classifiers should be applied.

Figure 24 – Simple perceptron accuracy boxplot



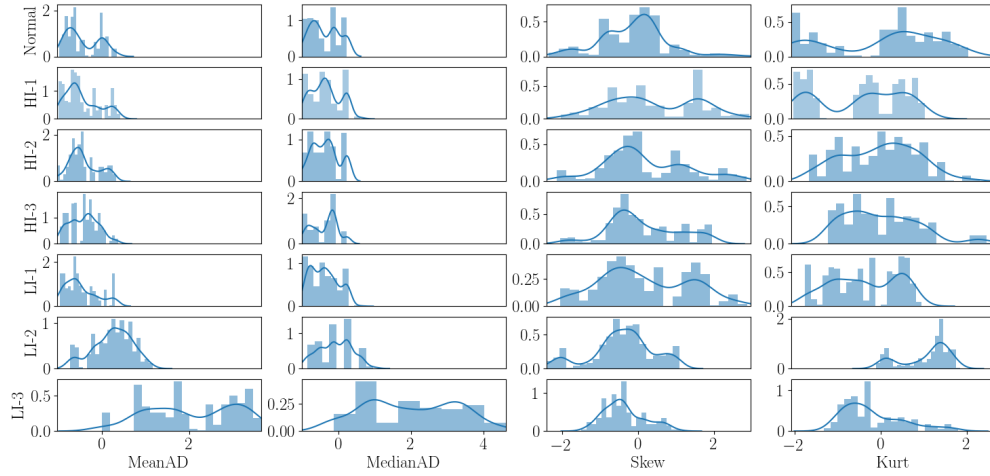
Source: Author

4.2.2 Bayesian classifiers

The Bayesian classifiers assume the data vector attributes are random variables with Gaussian problem density. Thus, the first step to Bayesian classifier evaluation is to verify if the distribution of the data fits this assumption. In Figure 25, the histograms of the features per class are displayed. The premise the features follow a Gaussian distribution is completely disproved. However, there are situations that even when this premise does not hold, Bayesian classifier achieves good results and it was analyzed.

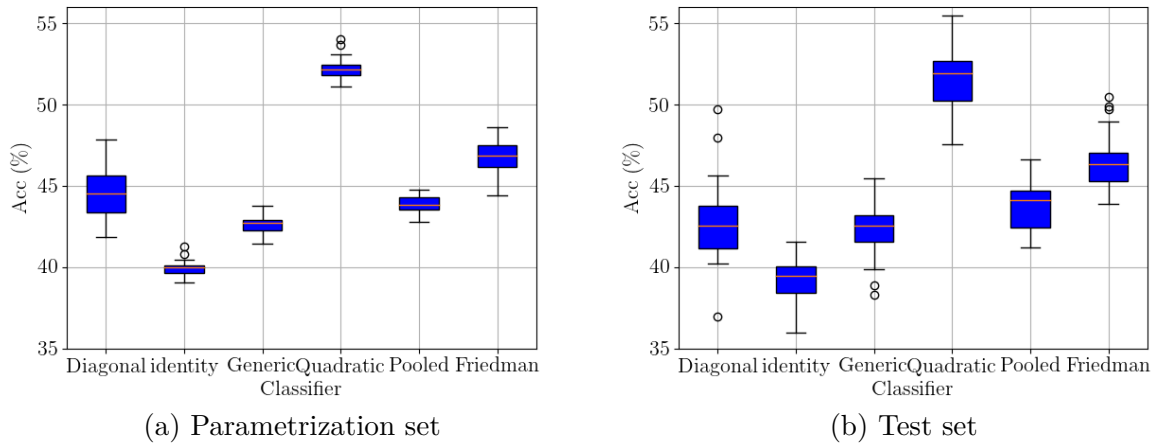
There are variations on the classifiers based on different estimates of the covariance matrix. These classifiers are non parametric, except by the λ factor of Friedman's estimate. λ was fixed in 0.5, to operate between the quadratic and linear discriminant function. For the parameterization, 80 % of the dataset was used, and 20 % for testing. In Figure 26 there are the accuracy box-plots for all the variations of the covariance matrices applied. In Figure 26a, the results for parameterization sets are presented and in Figure 26b, for the testing sets. As expected, the quadratic discriminant function presented the best result, about 52 % accuracy in the testing set. It represents a significant improvement in comparison to simple perceptron, even considering the results of the other covariance matrices estimates.

Figure 25 – Features histogram per class



Source: Author

Figure 26 – Bayesian classifiers accuracy box-plots for parameterization and test data sets, with different co-variance matrices



Source: Author

The features distribution among classes explain the poor results obtained using Bayesian Classifiers. This way, it is necessary to evaluate the performance of other classifiers.

4.2.3 Other classifiers

Using the features and procedures described before, the MLP, RBF, ELM, KNN, SVM and OPF classifiers were applied and their performance compared. For MLP, a grid search was performed to find the best combination of number of hidden neurons

$N_h=[5, 10, 15, 20, 30, 40, 50, 60]$, number of epochs of training $N_e=[500, 1000, 1500, 2000, 3000]$ and initial learning rate for hidden weights $\eta = [0.5, 0.3, 0.1, 0.05, 0.001]$. The best results were achieved with $N_h = 50$, $N_e = 2000$ and $\eta = 0.05$. This last parameter decays exponentially to 0.0001. The learning rate of the output layer weights is smaller than the hidden layer, $\frac{\eta}{3}$, to try to equalize the neurons learning speed in different layers, because the last layers usually have larger local gradients than the layers at the front end of the network (HAYKIN, 2008). The weights are adjusted using the gradient descent method and back-propagation of the error, using the iterative or stochastic methods. The activation function for all neurons is hyperbolic tangent and bias equal to 1. For the target values, +0.98 to the correct class and -0.98 for the remaining, instead the asymptotic limits, to avoid the hyperbolic tangent saturation region and prevent the weights to assume high absolute values (HAYKIN, 2008).

For ELM, the activation function is also hyperbolic tangent. The hidden weights are randomly assigned and the number of hidden units is selected using grid search and 10-fold cross validation, with $N_h = [10, 20, 30, \dots, 90, 100, \dots, 300, 310, 320, 330]$. The output neurons were analytically defined using LMS. The best results were achieved with $N_h = 310$. This value is extremely high and methods to reduce the number of hidden neurons should be applied to improve the generalization capacity of the network.

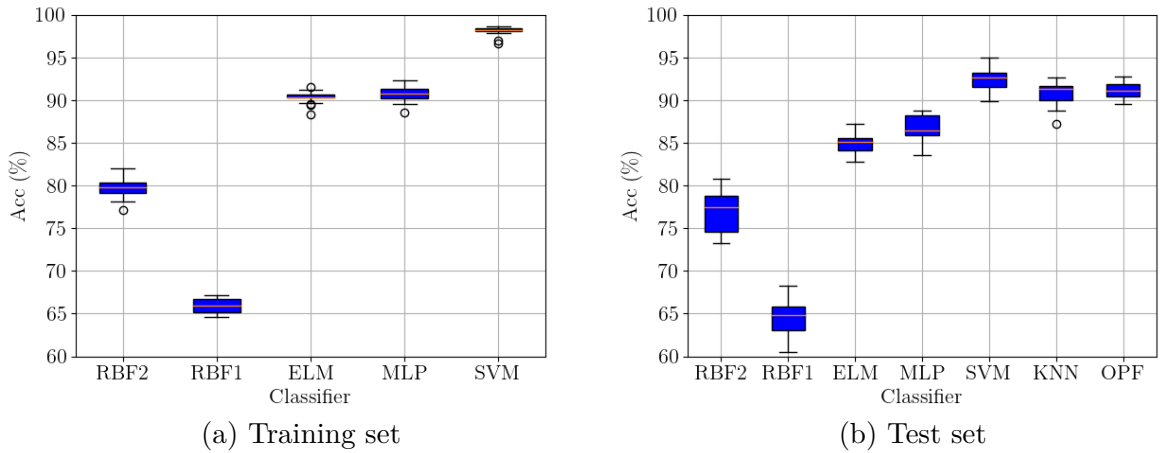
Two different approaches were used for training the RBF neural network. In the first one, referenced as RBF_1 , the number of centroids and the spread σ were selected using grid search, in the following space: $N_c=[5, 10, 20, 30, 40, 50, 60]$, $\sigma=[0.3, 0.7, 1, 1.2, 1.5, 2, 2.5, 3]$. The centroids position were defined using the k-means algorithm, an unsupervised approach. The best results were achieved with $N_c = 30$ and $\sigma = 2$. In the second approach, RBF_2 , the k-means in class-specific clustering approach was applied and, after the centroids positions were defined, the spread was calculated using Equation 2.49. In this case, the only parameter defined using grid search was the number of centroids per class $N_{cc}=[5, 10, 15, \dots, 50, 55]$. After the execution of k-means, the centroids which none training patterns associated are removed, resulting in a total number of centroids much inferior to the chosen initially. The best result was achieved for $N_{cc} = 50$, with the average total number of centroids 86.35 ± 4.18 and the $\sigma = 0.5 \pm 0.02$. This approach presented better accuracy results than RBF_1 . For both approaches, output neurons weights were defined using LMS.

In KNN, the Euclidean Distance was used and grid-search to determine the number of neighbors $k \in [1, 40]$ was performed. The best result was achieved with 1 neighbor. For SVM, the RBF kernel was used and the parameters σ and c were defined using grid search, where $\sigma=[1.5, 1, 0.25, 0.1, 0.001]$ and $C=[1, 10, 100, 1000, 10000, 100000]$. The best result was achieved using $c=10000$ and $\sigma=1$, which resulted in 642.25 ± 24.2 support vectors. It corresponds to about 32% of the training set size. The only parameter to be defined

to OPF classifier is the distance metric, which the following were evaluated: Euclidean, Manhattan, Chi Squared, Canberra, Squared Chord, Squared Chi Squared and Bray Curtis. Manhattan was the best.

In Figure 27 there is the global accuracy box-plot comparison of these classifiers in training and test data sets. SVM, OPF and KNN, respectively, presented the best results. For SVM, the global accuracy on the test set was between 90 % to 95 %, meanwhile OPF and KNN presented very similar results, between 89 % to 93 %. MLP and ELM were worse, ranged from about 82 % to 89 % and RBF presented results quite lower, about 60 % to 68 % for configuration 1 and 74 % to 81 %, for configuration 2. The results for training set followed similar behavior, despite being slightly higher, as expected. KNN and OPF training set accuracy were not presented because it would be 100 %, since all the training patterns are used in prediction.

Figure 27 – Comparison of average accuracy among the evaluated classifiers



Source: Author

RBF classifier will not be considered for the next experiments because its considerably lower performance compared to SVM, KNN, OPF and MLP. ELM will also be eliminated due its huge amount of neurons in hidden layer which achieved results close to MLP with significantly less hidden neurons.

4.2.4 Relevance of MeanAD and MedianAD

The similar behavior presented by the features MeanAD and MedianAD, in the pair-plots of Figure 23 and 25, raises the questioning about the relevance of having both variables in the feature set. To clear this point, the average confusion matrix of the model that achieved the best results (SVM) is compared to the results without the MedianAD in Tables 1 and 2. The removal of the MedianAD decreases the average accuracy from 92.42

% to 90.02 % and it also significantly decreases the Normal Class True positive rate, from 94.42 % to 91.43 %. It contributes to decrease the credibility of the system, because more normal patterns would be misclassified as failures ones, generating false alarms. This is very serious to industrial applications, because it can cause unnecessary interruptions of production to maintenance. The Table 3 summarizes the results for training and testing sets. For all of them, the inclusion of MedianAD impacts positively on the accuracy, despite the lower impact for KNN and OPF. This way, the MedianAD should continue on the feature set. The KNN and OPF training set accuracy is not presented because those classifiers use all the training patterns in the prediction and this way, they always get the classification of those patterns right. Now on, the feature set composed by skewness, kurtosis, meanAD and medianAD is addressed as Original FS.

Table 1 – Test set confusion matrix for SVM and Original FS (MeanAD, MedianAD, Skewness and Kurtosis)

Actual/Predicted	Normal	HI-1	HI-2	HI-3	LI-1	LI-2	LI-3
Normal	94.42	2.004	1.861	1.145	0	0.5727	0
HI-1	2.178	88.98	0.6601	1.188	6.997	0	0
HI-2	1.713	1.186	86.5	6.588	3.755	0.2635	0
HI-3	1.164	1.1	5.692	91.46	0.5821	0	0
LI-1	0.1313	6.96	4.137	0.6566	88.12	0	0
LI-2	0.4868	0	0.6954	0	0	97.64	1.182
LI-3	0	0	0	0	0	1.663	98.34

Table 2 – Test Set confusion matrix for SVM using MeanAD, Skewness, Kurtosis as features

Actual/Predicted	Normal	HI-1	HI-2	HI-3	LI-1	LI-2	LI-3
Normal	91.43	5.447	1.67	1.162	0.07262	0.2179	0
HI-1	3.6	88.2	0.8667	0.9333	6.4	0	0
HI-2	3.928	1.683	79.93	6.92	6.983	0.5611	0
HI-3	1.007	1.275	5.235	92.28	0.06711	0.1342	0
LI-1	0	10.82	3.43	0.4617	85.29	0	0
LI-2	0.7725	0	1.334	0	0	96.28	1.615
LI-3	0	0	0	0	0	1.838	98.16

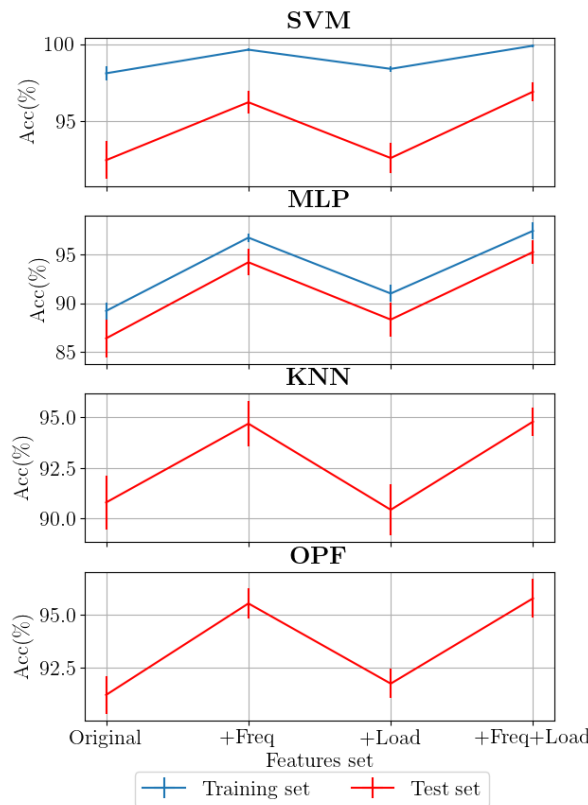
Table 3 – Average global accuracy comparison between Original FS and after medianAD is removed

	Original FS				Without MedianAD			
	Training		Testing		Training		Testing	
	Acc (%)	Std	Acc (%)	Std	Acc (%)	Std	Acc (%)	Std
SVM	98.13	0.47	92.42	1.2	96.56	0.22	90.02	1.2
MLP	89.2	0.89	86.38	1.95	86.99	1.08	84.45	1.91
OPF	-	-	91.23	0.88	-	-	91.02	1.17
KNN	-	-	90.69	1.2	-	-	90.37	0.95

4.3 Evaluation of frequency and load as additional features

Other two information already present in the dataset and that can be used as features are motor frequency and load. An analysis about the inclusion of those features is performed using the four classifiers with best results in the previous analysis: SVM, KNN, MLP and OPF. In Figure 28 there are the accuracy results for all the classifiers considering the insertion of each feature individually and combined. The incorporation of frequency increases significantly the accuracy for all classifiers. However, the inclusion of load information does not contribute significantly to improve the classification accuracy. Even with the small contribution of load individually, the best results achieved include both information.

Figure 28 – Accuracy comparison of different features set and classifiers



Source: Author

Table 4 synthesizes the accuracy comparison between the results with and without the load. If load is included or not, the true positive rate increases for some classes, meanwhile decreases for others, keeping the global accuracy almost equivalent. Despite similar hit rate presented by all the classifiers, further considerations can be made analyzing the confusion matrices of both possibilities presented in tables 5 to 10.

Table 4 – Comparison between mean global accuracies with and without the load

	Frequency + Load				Frequency			
	Training		Testing		Training		Testing	
	Acc (%)	Std	Acc (%)	Std	Acc (%)	Std	Acc (%)	Std
SVM	99.94	0.1	96.94	0.64	99.68	0.11	96.25	0.75
MLP	97.43	0.87	95.25	1.21	96.73	0.45	94.2	1.36
OPF	-	-	95.76	0.92	-	-	95.52	0.71
KNN	-	-	94.79	0.7	-	-	94.69	1.14

With the load included, the highest global accuracy is achieved by SVM (96.94 %), but the maximum true positive rate of normal class is achieved with MLP (98.71 %). The highest TP rate for each type of failure class is achieved by SVM, what explains the best global accuracy. OPF stands out because it achieves very low false negative rate: the only defective patterns classified as normal are HI-1 and HI-2 conditions, the lightest fault types. Without the load, the best global accuracy is also achieved by SVM (96.25 %). It also presented the highest sensitivity for each class, including the normal (98.65 %), that is even higher than considering the load for the same classifier. The amount of failure patterns misclassified as normal is similar to all the classifiers. Thus, for the situation without load information, SVM seems to be the best choice, considering the best global accuracy and sensitivity.

With or without the load, it is noticeable the confusion between classes HI-1 and LI-1 for all the classifiers. It has also been discussed by Vieira, Medeiros and Silva (2016) using motor current signature analysis. Another interesting point is that the misclassification percentage between HI-2 and Normal class is more significant than HI-1 errors, what is counterintuitive considering failure severity. Despite these considerations, the detection of the problem is clearly more effective as the failure intensity increases (considering the same failure types high or low impedance), as expected.

Table 5 – Test set confusion matrix for SVM and Original FS + frequency + load

Actual/Predicted	Normal	HI-1	HI-2	HI-3	LI-1	LI-2	LI-3
Normal	98.12	0.2174	0.5797	1.087	0	0	0
HI-1	0.3236	95.21	0	0.4531	4.013	0	0
HI-2	1.501	0.1305	94.32	0.8486	2.742	0.4569	0
HI-3	0.6285	1.257	0.7542	97.36	0	0	0
LI-1	0.271	2.913	2.1	0.06775	94.65	0	0
LI-2	0.2158	0	0	0	0	99.57	0.2158
LI-3	0	0	0	0	0	0.2046	99.8

Table 6 – Test set confusion Matrix for SVM and Original FS + frequency

Actual/Predicted	Normal	HI-1	HI-2	HI-3	LI-1	LI-2	LI-3
Normal	98.65	0.2132	0.4264	0.7107	0	0	0
HI-1	0.5913	93.89	0.0657	0.3942	5.059	0	0
HI-2	1.248	0.4596	92.58	1.641	3.152	0.9192	0
HI-3	0.9855	1.183	0.3942	97.11	0.3285	0	0
LI-1	0.1983	4.494	2.313	0.4627	92.53	0	0
LI-2	0.1404	0	0.2809	0	0	99.58	0
LI-3	0	0	0	0	0	0.06807	99.93

Table 7 – Test set confusion Matrix for OPF and Original FS + frequency + load

Actual/Predicted	Normal	HI-1	HI-2	HI-3	LI-1	LI-2	LI-3
Normal	97.06	0.6289	1.747	0.559	0	0	0
HI-1	0.1323	91.53	0.2646	0.463	7.606	0	0
HI-2	1.069	0.334	94.19	1.67	1.67	1.069	0
HI-3	0	1.178	1.571	97.12	0.1309	0	0
LI-1	0	7.26	0.7848	0.327	91.63	0	0
LI-2	0	0	0.3499	0	0	99.65	0
LI-3	0	0	0	0	0	0.3487	99.65

Table 8 – Test set confusion Matrix for OPF and Original FS + frequency

Actual/Predicted	Normal	HI-1	HI-2	HI-3	LI-1	LI-2	LI-3
Normal	96.23	0.5698	2.422	0.7123	0.07123	0	0
HI-1	0.2676	88.7	0.3344	0.602	10.1	0	0
HI-2	1.688	0.4545	91.75	3.312	1.883	0.9091	0
HI-3	0.1336	1.737	2.672	94.92	0.5344	0	0
LI-1	0.2601	7.607	1.235	0.3901	90.51	0	0
LI-2	0.07047	0	0.6342	0	0	99.22	0.07047
LI-3	0	0	0	0	0	0.2045	99.8

Table 9 – Test set confusion Matrix for MLP and Original FS + frequency + load

Actual/Predicted	Normal	HI-1	HI-2	HI-3	LI-1	LI-2	LI-3
Normal	98.71	0.0714	0.3571	0.3571	0.5	0	0
HI-1	0.2632	89.54	0.4605	0.9868	8.684	0.0658	0
HI-2	0.9868	0.5263	92.5658	3.224	2.1711	0.4605	0.0658
HI-3	0.3289	0.8553	1.579	96.84	0.0658	0.2632	0.0658
LI-1	0.1333	6.333	1.8	1.267	90.333	0	0.1333
LI-2	0.06944	0	0.06944	0	0	99.79	0.0694
LI-3	0	0	0	0	0	0.3472	99.65

Table 10 – Test set confusion Matrix for MLP and Original FS + frequency

Actual/Predicted	Normal	HI-1	HI-2	HI-3	LI-1	LI-2	LI-3
Normal	97.79	0.2143	0.7143	0.5714	0.3571	0.3571	0
HI-1	0.3947	88.6184	0.6579	0.5921	9.737	0	0
HI-2	1.184	0.7237	90.92	4.013	2.697	0.4605	0
HI-3	1.053	1.118	1.513	95.79	0.4605	0.0658	0
LI-1	0.6667	7.667	2.2	1.867	87.33	0.1333	0.1333
LI-2	0	0	0.0694	0	0	99.93	0
LI-3	0	0	0	0.0694	0	0.1389	99.79

Table 11 – Teste set confusion matrix for KNN and Original FS + frequency + load

Actual/Predicted	Normal	HI-1	HI-2	HI-3	LI-1	LI-2	LI-3
Normal	96.72	0.7671	1.883	0.6276	0	0	0
HI-1	0.7175	90.08	0.3262	0.7828	8.089	0	0
HI-2	1.445	0.3283	93.3	2.035	1.904	0.9849	0
HI-3	0.3378	1.622	1.892	95.88	0.2703	0	0
LI-1	0.197	8.798	0.8536	0.6566	89.49	0	0
LI-2	0	0	0.8158	0	0	99.12	0.06798
LI-3	0	0	0	0	0	0.4237	99.58

Table 12 – Test set confusion matrix for KNN and Original FS + frequency

Actual/Predicted	Normal	HI-1	HI-2	HI-3	LI-1	LI-2	LI-3
Normal	97.39	0.5806	1.451	0.508	0.07257	0	0
HI-1	0.3369	89.56	0.5391	0.7412	8.827	0	0
HI-2	1.7	0.0654	92.02	3.336	1.504	1.373	0
HI-3	0.3979	1.989	1.459	96.02	0.1326	0	0
LI-1	0.2562	8.008	0.9609	0.8328	89.94	0	0
LI-2	0.2063	0	0.7565	0	0	99.04	0
LI-3	0	0	0	0	0	0.4775	99.52

In this dataset, the drive frequency and load data were manually registered. Considering a real application, the frequency can be read from the frequency converter that fed the motor, using one of its analog outputs. However, the load information does not increase significantly the performance and it is also harder to obtain, because it would imply in measuring other motor variables, as operating speed, increasing the cost of the system. Thus, it will not be considered in the rest of this study.

When performing a further analysis of the SVM model confusion matrix including only the frequency as a new feature, the following points are observed:

- Normal patterns are confused with high impedance failures only. However, there is a small amount of low impedance patterns of types 1 and 2 misclassified as normal.

- LI-3 failure is always detected, despite the insignificant percentage (0.06 %) of patterns classified as LI-2.
- Considering the grouping of failures in accordance with type of failure (high and low impedance), which results in 3 classes, the accuracy is $96.86\% \pm 0.79$. The reason to the similar accuracy value is that a significant amount of classification errors happens between high and low impedance classes, specially HI-1 and LI-1, as already highlighted. The confusion matrix for this representation is on Table 13.
- If the failures are grouped by the amount of shorted turns, there are 4 classes and the accuracy is $97.85\% \pm 0.58$. The accuracy improves because there is significant confusion between failures with the same number of shorted turns. Again, HI-1 and LI-1. The confusion matrix for this representation is in Table 14.
- Considering only normal and failure classes, $99.35\% \pm 0.35$ accuracy is obtained, with a failure false negative rate of 0.5 % and failure false positive rate of 1.35 %.

A similar analysis for MCSA, about grouping the failures classes by the type or the number of shorted turns, is also described by Vieira, Medeiros and Silva (2016).

Table 13 – SVM test set confusion matrix grouped by failure type

Actual/Predicted	Normal	HI	LI
Normal	98.65	1.35	0
HI	0.9415	95.91	3.153
LI	0.1135	2.587	97.3

Table 14 – SVM test set confusion matrix grouped by amount of short turns

Actual/Predicted	Normal	Level 1	Level 2	Level 3
Normal	98.65	0.2132	0.4264	0.7107
Level 1	0.3954	97.99	1.186	0.4283
Level 2	0.7126	1.866	96.57	0.8483
Level 3	0.5015	0.769	0.234	98.5

Table 15 – SVM test set confusion matrix with all the failures grouped

Actual/Predicted	Normal	Failure
Normal	98.65	1.35
Failure	0.5349	99.47

4.4 Conclusion

In this chapter, it has been presented the process to design a model to detect and classify the type of stator winding inter-turn short-circuit using the features: mean absolute

deviation, median absolute deviation, skewness, kurtosis and frequency, calculated directly from the voltage signal induced by the motor axial flux. The data were acquired from a copper coil placed on the motor shaft. Several classifiers were experimented: Simple Perceptron, different configurations of Bayesian classifiers, OPF, SVM, MLP, ELM, RBF and KNN. Among all the classifiers tried, MLP, SVM, KNN and OPF presented good and similar results, about 95 % accuracy.

The best result was achieved using SVM (96.25 % accuracy for seven classes and 99.35 % for two classes). Those are encouraging results, because using only five statistical features, calculated directly from the signal in time domain, it is possible to detect the failures with low FN rate, about 0.5 % and low FP rate, 1.35 %. Furthermore, even very incipient failures are detected, only 1.41 % of shorted turns, with a TP rate over 92.5 %.

A similar study using MCSA with Fourier transform and MLP, with the same approach of short-circuit emulation and the same composition of the dataset in relation to the failures subtypes, frequency and load obtained about 80 % accuracy, considering only normal and failure classes. The same authors, using PCA and simple perceptron, obtained about 98 % hit rate (OLIVEIRA, 2014). In the present study, using only statistical features calculated directly from the flux signal in the time domain and frequency information, better accuracy results were achieved, what seems to indicate that using flux information is a better approach to detect this type of failure than MCSA, as suggested by other researches (YAZIDI et al., 2004). A similar study was performed using flux signal and the induction machine as a generator, with the features skewness, kurtosis, RMS value and variance, obtaining 95.62 % accuracy for 7 classes, using Bayes classifier. To compose the features set, the voltage on the DC-bus, generator's frequency and the RMS current were included (SOUSA et al., 2018). Another indication of the relevance of leakage flux signal in this field of study.

The inclusion of load in feature set improves the performance of the models, but due the requirements for obtaining this information, it is not going to be considered in the proposed approach, at least for now. In the next chapter is described a study to further improve the detection of short-circuit failures using axial flux.

5 The inclusion of multiresolution decomposition in features extraction

In this chapter, the inclusion of the DWT to decompose the flux signals in independent frequency bands during the feature extraction phase is described. Different levels of decomposition are evaluated using KNN, SVM, MLP and OPF classifiers. To perform feature subset selection (FSS), analysis of variance is applied.

5.1 Wavelet Transform applicability and its advantages

Spectral analysis is one of the techniques often applied to induction motor incipient failure detection, using the conventional FFT. However, it requires long sampling periods to achieve high-frequency resolution. Another disadvantage of using FFT is the assumption the signal is stationary, because in case of transient and discontinuities, it will be spread over the whole frequency spectrum. The usage of STFT addressed some of the drawbacks of the FT, but other remain unsolved, like the usage of a fixed window, what result in fixed resolution for both frequency and time (SIFUZZMAN; ISLAM; ALI, 2009). In contrast, WT uses short windows at high frequencies and long windows at low frequencies, resulting in good time resolution at high frequencies, while the frequency resolution becomes good at low frequencies. This is the principle of the multiresolution analysis.

In the previous analysis, the signal in time domain was used to calculate some statistical features: MeanAD, MedianAD, kurtosis and Skewness. In this scenario, all frequency components present in the signal influences the feature values and, consequently, the classification results. Using the Discrete Wavelet Transform it is possible to analyze a signal at different frequency ranges with different resolutions. This way, it is possible to calculate the statistical features by frequency bands, removing some frequency range components that might be disturbing the classification and also increasing the feature space dimensionality.

5.2 DWT initial exploration

The sampling frequency used to acquire the flux signal was 10 kHz. Previous research in the same field indicates that the main frequency components related to short-circuit failures are lower than 1 kHz for 60Hz driving frequency (J.PENMAN, 1994; FROSINI et al., 2012). Based on this information, as a starting point, 2 levels of decomposition have been chosen, splitting the original signal in wavelet signals with frequency sub-bands

according to Table 16.

Table 16 – Frequency bands for two levels wavelet decomposition

Level	Approximation		Detail	
0 (Original signal)	0-5000 Hz			
1	A_1	0-2500 Hz	D_1	2500-5000 Hz
2	A_2	0-1250 Hz	D_2	1250-2500 Hz

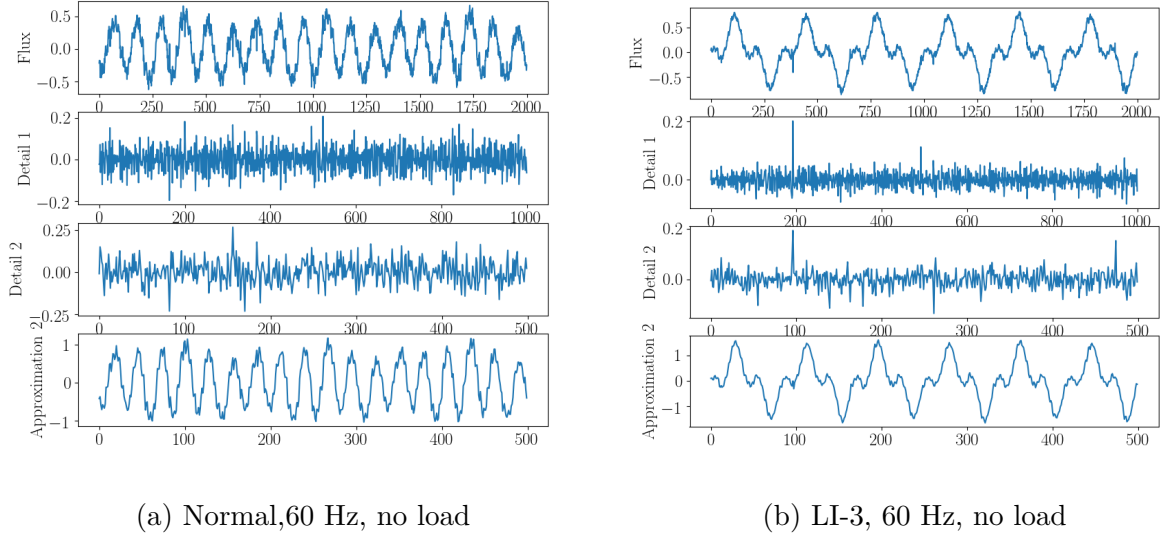
Before the application of the DWT, it is necessary to select the mother wavelet. There are no well defined rules to make this choice. It depends upon the type of signal to be analyzed and application. Among the several types, Daubechies mother Wavelet family is a common choice due its simplicity. Another parameter to be decided is the order of the wavelet. There exists a trade-off between the order of the wavelets and the computation time. Higher order wavelets are smoother and are better able to distinguish between the various frequencies, but require more computation time (MERRY, 2005). For this reason, a second order Daubechies was chosen to start the investigation.

The approximation and detail coefficients of the 2 level decomposition using the db-2 mother wavelet are presented in Figure 29. The motor is healthy in Figure 29a, and in Figure 29b, the most severe condition LI-3 is applied. It is easy to notice the detail coefficients contain the high frequency components of the flux signal and the approximation corresponds to the low frequency band, as described in Table 16. Level 2 approximation coefficients A_2 , are similar to the signals in time domain after a low-pass filtering for both examples. However, the detail coefficients D_1 and D_2 have low amplitude compared to A_1 and are similar in the examples, despite the signals represents very different conditions. It seems to indicate those coefficients are not useful to detect the condition monitored. This is already expected, considering the flux signal has been analog filtered with 1 kHz cutoff frequency before its acquisition.

5.2.1 Features set exploration using KNN

The study about the inclusion of frequency and load in the feature set, described in section 4.3, indicated the impact in the global mean accuracy of a new feature insertion follows a trend curve, despite the classifier and its accuracy results, as shown in Figure 28. Due its implementation simplicity and classification effectiveness, KNN is commonly used as a standard classifier to evaluate and compare the performance of different feature sets (WANG et al., 2015). Based on it, in a first moment, the study about the impact of the decomposition of the flux signal in two levels using the DWT, before the statistical features calculation is performed using KNN only. Then, with the best feature sets, the analysis is extended to other classifiers used in the previous study. The same rules described in Section 4.2 are applied to all the classifiers training.

Figure 29 – Flux approximation and detail coefficients for 2 level DWT using Db-2



Source: Author

The statistical features skewness, kurtosis, mean absolute deviation and median absolute deviation used before are calculated for the approximation and detail coefficients obtained after the multiresolution decomposition. Also, another feature is added, relative wavelet energy, described in 2.2.2.2.1. The hypothesis investigated is if the removal of higher order frequency bands will improve the detection.

Table 17 describes the feature sets analyzed and labels each one. D_1 and D_2 are, respectively, the detail coefficients of level 1 and 2, and A_2 is the level 2 approximation coefficient. In FS1, the features are calculated for all the coefficients D_1 , D_2 and A_1 . In FS2, only D_2 and A_2 are used. In FS3, only A_2 . In FS4, the frequency is included and in FS5, RWE (Relative Wavelet Energy) is removed to evaluate its relevance in the classification accuracy.

Three metrics were evaluated: 7 classes mean global accuracy, binarized mean accuracy and normal class TP rate. The second metric groups all the failures together after the classification task has been executed. It represents the model's ability to detect the failures, independent of identifying its severity. This metric is used because, despite distinguishing the failure intensity is an objective of this investigation, the most important task is to detect the failures, even the most incipient ones, which only 1.41 % of shorted turns. The normal class TP rate is evaluated because, as already explained in the last chapter, it is very important to the system reliability it does not create false alarms. Figure 30 display the results of those three metrics for all the features sets described on Table 17. The 7 classes accuracy for each FS is compared to the result without using the DWT,

represented by the black dashed line (FS0). The vertical bar of the curves represents the standard deviation.

Table 17 – Features set description

Feature Set	Description	Number of features
FS1	Mean absolute deviation, median absolute deviation, kurtosis, skewness and relative energy for D_1, D_2 and A_2 .	15
FS2	Mean absolute deviation, median absolute deviation, kurtosis, skewness and relative energy for D_2 and A_2 .	10
FS3	Mean absolute deviation, median absolute deviation, kurtosis, skewness and relative energy for A_2 .	5
FS4	Mean absolute deviation, median absolute deviation, kurtosis, skewness and relative energy for A_2 and frequency.	6
FS5	Mean absolute deviation, median absolute deviation, kurtosis and skewness for A_2 and frequency	5

This experiment demonstrated the features calculated from the Detail coefficients of level 1 and 2 do not contribute to the detection of failure and make the accuracy worse (represented by features set 1 and 2 of Figure 30), as already expected. The classification performance using features calculated from A_2 , the approximation coefficient of the second level of decomposition, (FS3) is better than when the features are calculated from the original signal directly. The frequency information still relevant, because all the metrics evaluated increases when it is added (FS4), and the RWE also demonstrated its importance, because its removal worsens all the metrics. Considering KNN and the best features set, FS4, the average global 7 classes accuracy improved from 94.69 ± 1.14 with FS0 to 97.26 ± 0.55 . For healthy and failure classes only, the false positive rate decreased from 2.61 % to 1.48 % and false negative from 0.49 % to 0.29 %.

Table 18 presents the confusion matrix for this new approach. To check the improvements in classification between failure types, compare it to Table 12. The confusion among defective classes decreased, in general, with emphasis on the misclassification between HI-1 and HI-2, which has been significantly reduced. Furthermore, none low impedance failure patterns were wrongly classified as normal ones. The opposite is also true.

5.2.2 Results with other classifiers

The classifiers SVM, MLP and OPF were applied to FS4 and also had their accuracy results improved. They are displayed on Table 19 for 7 classes and 2 classes (binarized).

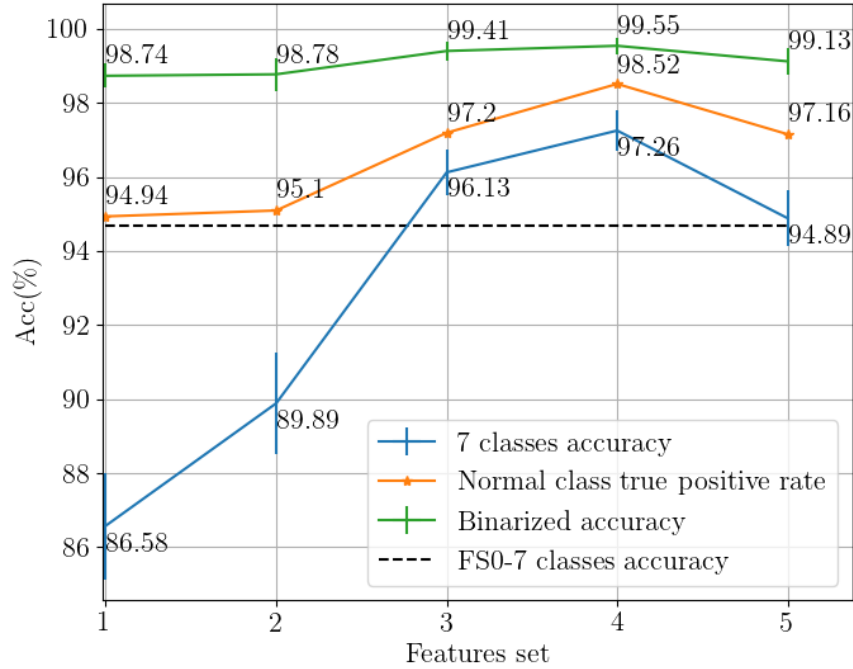


Figure 30 – Comparison of different features set analyzed with KNN for 2 levels of decomposition

Table 18 – Test set confusion matrix for FS4 + KNN

Real/Predicted	Normal	HI-1	HI-2	HI-3	LI-1	LI-2	LI-3
Normal	98.52	0.6347	0.07052	0.7757	0	0	0
HI-1	0.3979	93.83	1.724	0.06631	3.979	0	0
HI-2	1.084	1.423	95.05	1.558	0.4743	0.4065	0
HI-3	0.2606	0.7166	0.456	98.57	0	0	0
LI-1	0	3.706	0.5852	0.3251	95.38	0	0
LI-2	0	0	0	0	0	100	0
LI-3	0	0	0	0	0	0.2742	99.73

SVM presented the highest accuracy in the testing set, 98.59 % for 7 classes and 99.84 % for 2 classes, followed by OPF, MLP and KNN.

In Table 20 there are the average binary confusion matrices for the 4 classifiers, while in Tables 21 to 23 there are the complete confusion matrices for SVM, OPF and MLP. SVM also presented the lowest false positive rate (0.2847 %) and false negative rate (0.1452 %), which is pretty much even to OPF, with 0.1339 %. The miss classification between types of failures also improved for all the classifiers. For SVM, the lowest sensitivity happens to HI-1, 95.81 %, which is the most difficult failure type to detect due its similarity to LI-1 and normal patterns. This result suggests this approach as a good option not only to detect the short-circuit but also to track its evolution, considering the distinction between failure types.

Table 19 – Classification results for FS4

	Training		Testing			Training		Testing	
	Acc(%)	Std	Acc(%)	Std		Acc(%)	Std	Acc(%)	Std
SVM	99.72	0.06	98.59	0.48	SVM	100	0	99.84	0.15
MLP	99.21	0.25	97.83	0.76	MLP	99.9	0.07	99.6	0.25
OPF	-	-	97.87	0.9	OPF	-	-	99.68	0.28
KNN	-	-	97.26	0.55	KNN	-	-	99.54	0.24

(a) 7 classes mean global accuracy

(b) Binarized mean global accuracy

Table 20 – Test set binary confusion matrix for FS4

SVM			MLP		
Real/Predicted	Normal	Failure	Real/Predicted	Normal	Failure
Normal	99.72	0.2847	Normal	98.14	1.857
Failure	0.1452	99.85	Failure	0.1678	99.83

KNN			OPF		
Real/Predicted	Normal	Failure	Real/Predicted	Normal	Failure
Normal	98.52	1.481	Normal	98.5	1.5
Failure	0.2908	99.71	Failure	0.1339	99.87

Table 21 – Test set confusion matrix for FS4 + SVM

Real/Predicted	Normal	HI-1	HI-2	HI-3	LI-1	LI-2	LI-3
Normal	99.72	0.07117	0	0.2135	0	0	0
HI-1	0.5151	95.81	0.06439	0	3.606	0	0
HI-2	0.2727	0.3408	98.09	0.9543	0	0.3408	0
HI-3	0.06398	0.7678	0.2559	98.91	0	0	0
LI-1	0	1.638	0.5461	0	97.82	0	0
LI-2	0	0	0	0	0	100	0
LI-3	0	0	0	0	0	0	100

Table 22 – Test set confusion matrix for FS4 + OPF

Real/Predicted	Normal	HI-1	HI-2	HI-3	LI-1	LI-2	LI-3
Normal	98.5	0.2857	0	1.214	0	0	0
HI-1	0.3927	94.7	0.8508	0.3272	3.73	0	0
HI-2	0.4005	1.268	96.53	1.202	0.1335	0.4673	0
HI-3	0	0.4575	0.3268	99.22	0	0	0
LI-1	0	3.219	0.4024	0	96.38	0	0
LI-2	0	0	0	0	0	100	0
LI-3	0	0	0	0	0	0	100

Table 23 – Test set confusion matrix for FS4 + MLP

Real/Predicted	Normal	HI-1	HI-2	HI-3	LI-1	LI-2	LI-3
Normal	98.14	0.1428	0.2857	1	0.3571	0.0714	0
HI-1	0.0658	96.38	0.6579	0.3289	2.5	0.0657	0
HI-2	0.1316	0.3947	97.76	0.7237	0.52634	0.4605	0
HI-3	0.4605	0.5922	0.5263	97.76	0.6579	0	0
LI-1	0.2	3	0.6	1	95.2	0	0
LI-2	0.1389	0	0	0	0	99.86	0
LI-3	0	0	0.0694	0	0	0	99.93

5.3 Increasing the levels of decomposition

In section 5.2 the study showed that the usage of level 2 approximation coefficients to calculate the statistical features improved the classification result. To analyze if a decomposition in more levels and an increase in the number of features can improve the results, a similar approach applied to section 5.2 is performed. Analysis of variance (ANOVA) will be used to eliminate features with low f-value score. The F-value scores the difference between the feature mean values for each class.

5.3.1 Feature set exploration using KNN

The results presented on Figure 30 indicated that the inclusion of the features calculated using D_1 and D_2 worsens the result while using A_2 only presented the best results. The wavelet transform Multiresolution Analysis (MRA) continues from this point, decomposing A_2 in more levels and eliminating the features of the highest frequency bands A_1 and A_2 . This study is divided into two stages, first the number of decomposition levels are gradually increased from 2 to 6 and the effect in classification is analyzed. Table 24 shows the frequency bands of the coefficients for each level of the MRA. At each level, the frequency resolution is increased. After establishing the optimum number of levels, some features are gradually removed, using ANOVA F-score, and again, the impact in classification is analyzed.

Table 24 – Frequency bands for the wavelet coefficients

Level	Approximation		Detail	
0 (Original signal)	0-5000 Hz			
1	A_1	0-2500 Hz	D_1	2500-5000 Hz
2	A_2	0-1250 Hz	D_2	1250-2500 Hz
3	A_3	0-625 Hz	D_3	625-1250 Hz
4	A_4	0-312.5 Hz	D_4	321.5-625 Hz
5	A_5	0-156.25 Hz	D_5	156.25-321.5 Hz
6	A_6	0-78.125 Hz	D_6	78.125-156.25 Hz

Table 25 describes the feature sets. For FS6, skewness, kurtosis, mean absolute deviation, median absolute deviation and WRE are calculated for both level 3 coefficients, D_3 and A_3 . Then, A_3 is decomposed in D_4 and A_4 and the same features are calculated for A_4 and D_4 , composing FS7, with the features calculated using D_3 . The same process is repeated to generate FS8 and FS9.

Figure 31 show the results for each feature set of Table 25. The increase in decomposition levels, decreases the average 7 classes global accuracy, compared to FS4, but improves the binary accuracy and normal class TP rate. That indicates the confusion between failure classes is increasing, while decreasing the miss classification between normal and failure. As already highlighted, the main purpose is to detect the failure and also keep the false positive rate low to increase reliability. This way, despite the decrease of 7 classes accuracy, the results showed that the increase of the decomposition levels improves the detection of the failure, compared to FS4. More levels of decomposition were not considered, because from FS8 to FS9, the accuracy for 7 classes and also normal TP rate have significantly diminished.

Table 25 – Features set description - increase in the decomposition levels

Feature Set	Description	Number of features
FS6	All features of D_3 and A_3 + frequency.	11
FS7	All features of D_3, D_4 and A_4 + frequency.	16
FS8	All features of D_3, D_4, D_5 and A_5 + frequency.	21
FS9	All features of D_3, D_4, D_5, D_6 and A_6 + frequency.	26

5.3.1.1 Feature subset selection

In the next phase of the exploration, FS6 (3 levels of decomposition), and FS8 (5 levels), that presented the best normal TP rate and binary accuracy, have some features eliminated, considering the hypothesis that some of them might not be very correlated to the problem and are prejudicing the classification. The one-way ANOVA is used to rank the features in accordance to its F-score, that indicates the predictor discriminative capability. The features with lower F - scores are gradually eliminated from the Feature set, in a backward elimination fashion, and its performance is evaluated using KNN, until no more improvements happen or it starts to get worse. This approach combines filtering and wrapper methods for feature subset selection, which speeds up the process and are less over-fitting prone than pure wrapper methods (DUCH, 2006).

5.3.1.1.1 Three levels of decomposition

The 11 features that compose FS6 (3 levels of decomposition) were ranked using F-score and the lowest ranked features were eliminated one by one. The FS sets at each

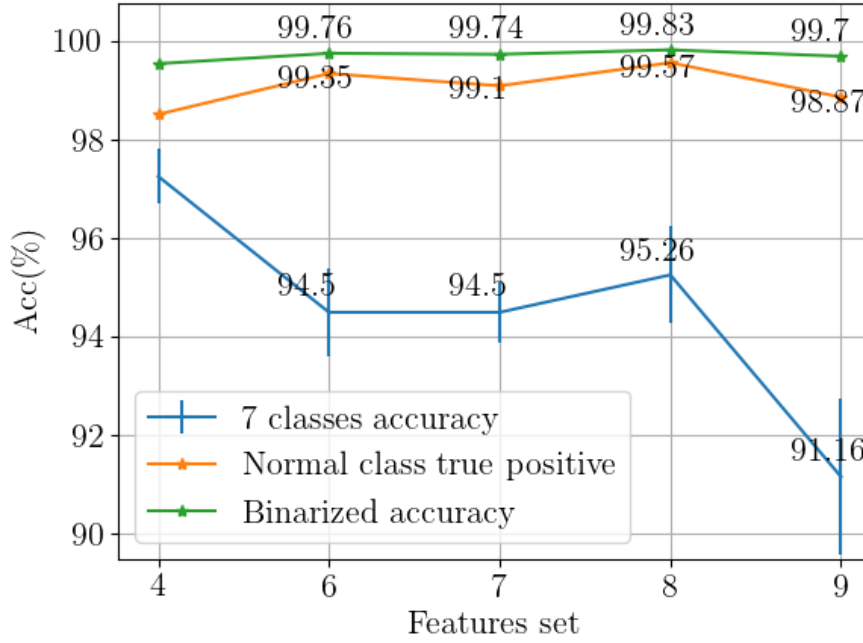


Figure 31 – Comparison of different features set (check Table 25 for the description), according to the number of DWT levels, with KNN

step of this feature subset selection process are described in Table 26. Figure 32 presents the accuracy results for 7 and 2 classes, and also the normal TP rate, similar to the previous graphics. Starting with FS6, after some features are removed (skewness A_3 and D_3 , kurtosis D_3), what corresponds to FS10 to FS12, the 7 classes average global accuracy increased from 94.5 % to 99.13 % and 2 classes accuracy statistically remained constant (from 99.76 % to 99.84 %). The average normal TP rate also increases from 99.35 % to 99.79 %. With FS13, the 7 classes accuracy decreases a little, to 99.03 %, but the normal TP rate achieves 100 %, with 99.9 % mean accuracy for 2 classes. After that, FS14, the three metrics get worse and the feature reduction process is over. Compared to FS4, the best result for 7 classes until now (which had 2 levels of DWT decomposition and 6 features), with one more level of decomposition and one more feature, FS13 represents an improvement of almost 2 % in 7 classes accuracy (from 97.26 % to 99.1 %) and solved the false positive error. One of the main concerns of this study.

In the next subsection, the same experiment is applied to FS8, to check if even better results can be achieved.

5.3.1.1.2 Five levels decomposition

A similar process followed to find the best sub feature set for 3 levels of decomposition was applied to FS8. Figure 33 presents the classification results of the features set described on Table 27. Despite some minor intermediate sags, when some features are

Table 26 – Features set description - selection using ANOVA for 3 levels of decomposition

Features set	Description	Number of features
FS10	Frequency, Mean AD A_3 , Median AD A_3 , Median AD D_3 , Kurt A_3 , Ep D_3 , Ep A_3 , MeanAD D_3 , Kurt D_3 , Sk D_3	10
FS11	Frequency, Mean AD A_3 , Median AD A_3 , Median AD D_3 , Kurt A_3 , Ep D_3 , Ep A_3 , MeanAD D_3 , Kurt D_3	9
FS12	Frequency, Mean AD A_3 , Median AD A_3 , Median AD D_3 , Kurt A_3 , Ep D_3 , Ep A_3 , MeanAD D_3	8
FS13	Frequency, Mean AD A_3 , Median AD A_3 , Median AD D_3 , Kurt A_3 , Ep D_3 , Ep A_3	7
FS14	Frequency, Mean AD A_3 , Median AD A_3 , Median AD D_3 , Kurt A_3 , Ep D_3	6

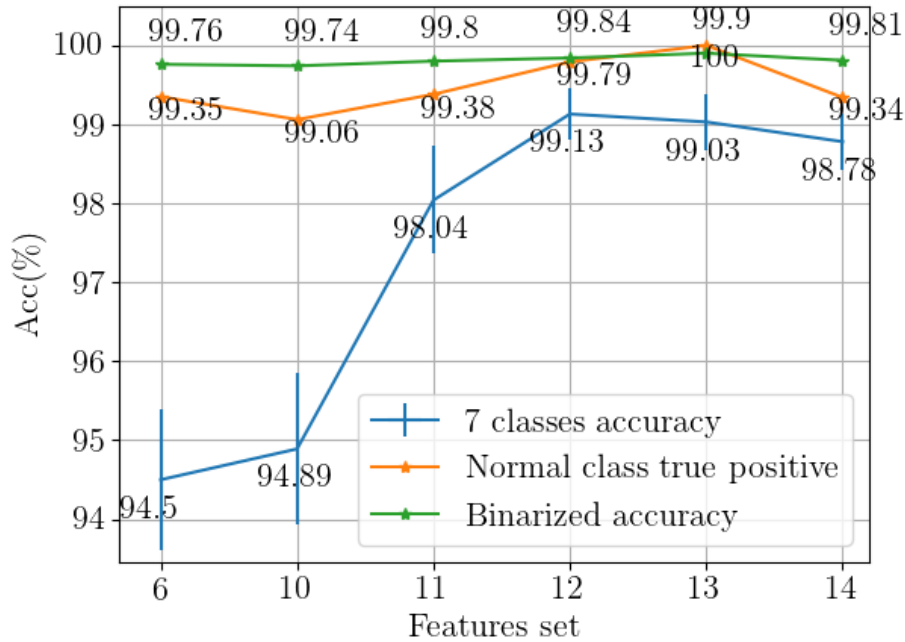


Figure 32 – Comparison of different features set (check Table 26 for the description), for 3 levels of decomposition with a KNN classifier

removed, from FS15 to FS20, the average 7 classes accuracy increases from 94.54 % to 98.52 % and, for 2 classes, from 99.81 % to 99.99 %. The average normal TP rate also increases from 99.5 % to 99.93 %. After this point, FS21 and FS22, the three metrics get worse and normal TP rate decreases to 99.07 %. Compared to FS13, which had 3 levels of DWT decomposition and 7 features, FS20, with 3 more levels of decomposition and the double of the features, presented lower 7 classes accuracy and normal TP rate and equivalent 2 classes accuracy. Despite very small, about 0.07 %, false positive cases still happen.

To analyze and compare better the classification performance obtained with FS13 and FS20, the results with other classifiers are presented and discussed in the next

Table 27 – Features set description - selection using ANOVA for 5 levels of decomposition

Features set	Description	Number of features
FS15	Frequency, Mean AD D_3 , Mean AD A_5 , Median AD D_3 , Median AD D_4 , Median AD D_5 , Median AD A_5 , Kurt D_5 , Kurt A_5 , Ep D_3 , Ep D_4 , Ep D_5 , Ep A_5 , Mean AD D_4 , Sk D_4 , Sk A_5 , Kurt D_3 , Mean AD D_5 , Sk D_3 , Median AD D_5	19
FS16	Frequency, Mean AD D_3 , Mean AD A_5 , Median AD D_3 , Median AD D_4 , Median AD D_5 , Median AD A_5 , Kurt D_5 , Kurt A_5 , Ep D_3 , Ep D_4 , Ep D_5 , Ep A_5 , Mean AD D_4 , Sk D_4 , Sk A_5 , Kurt D_3 , Mean AD D_5 , Sk D_3	18
FS17	Frequency, Mean AD D_3 , Mean AD A_5 , Median AD D_3 , Median AD D_4 , Median AD D_5 , Median AD A_5 , Kurt D_5 , Kurt A_5 , Ep D_3 , Ep D_4 , Ep D_5 , Ep A_5 , Mean AD D_4 , Sk D_4 , Sk A_5 , Kurt D_3 , Mean AD D_5	17
FS18	Frequency, Mean AD D_3 , Mean AD A_5 , Median AD D_3 , Median AD D_4 , Median AD D_5 , Median AD A_5 , Kurt D_5 , Kurt A_5 , Ep D_3 , Ep D_4 , Ep D_5 , Ep A_5 , Mean AD D_4 , Sk D_4 , Sk A_5 , Kurt D_3	16
FS19	Frequency, Mean AD D_3 , Mean AD A_5 , Median AD D_3 , Median AD D_4 , Median AD D_5 , Median AD A_5 , Kurt D_5 , Kurt A_5 , Ep D_3 , Ep D_4 , Ep D_5 , Ep A_5 , Mean AD D_4 , Sk D_4 , Sk A_5	15
FS20	Frequency, Mean AD D_3 , Mean AD A_5 , Median AD D_3 , Median AD D_4 , Median AD D_5 , Median AD A_5 , Kurt D_5 , Kurt A_5 , Ep D_3 , Ep D_4 , Ep D_5 , Ep A_5 , Mean AD D_4 , Sk D_4	14
FS21	Frequency, Mean AD D_3 , Mean AD A_5 , Median AD D_3 , Median AD D_4 , Median AD D_5 , Median AD A_5 , Kurt D_5 , Kurt A_5 , Ep D_3 , Ep D_4 , Ep D_5 , Ep A_5 , Mean AD D_4	13
FS22	Frequency, Mean AD D_3 , Mean AD A_5 , Median AD D_3 , Median AD D_4 , Median AD D_5 , Median AD A_5 , Kurt D_5 , Kurt A_5 , Ep D_3 , Ep D_4 , Ep D_5 , Ep A_5	12

subsection.

5.3.2 Results using other classifiers

The classifiers SVM, MLP and OPF were applied to FS13 and FS20. The accuracy results are summarized on Table 28. The binary confusion matrices for the four classifiers and the features set cited are presented on Table 29. Compared to FS04, the global 7 classes accuracy improved about 1 %, but the main improvements are the minimization of false negative and positive rates.

Considering the distinction between all the classes, the best result with the test set was achieved by the combination of FS20 and SVM, $99.23\% \pm 0.32$ accuracy. In this approach, on average, 99.92 % of the failures are detected and even very incipient defects achieve very high detection percentage, as exposed in the confusion matrix of Table 30. Only 0.46 % patterns of HI-2 are misclassified as normal samples, what corresponds to 0.08 % of failure patterns, a very low FN rate. Between failure classes, the confusion happens

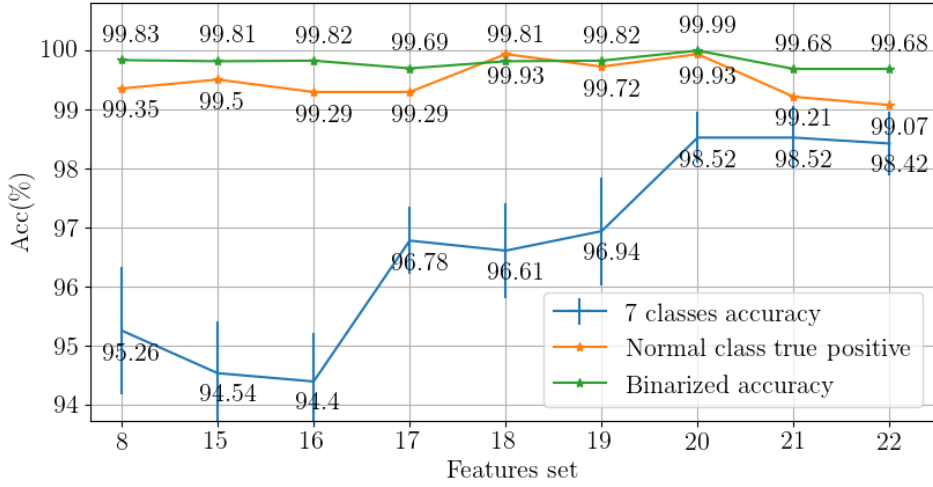


Figure 33 – Comparison of different features set (check Table 27 for the description), for 5 levels of decomposition with a KNN classifier

Table 28 – Classification results for FS13 and FS20

	FS13				FS20			
	Training		Testing		Training		Testing	
	Acc(%)	Std	Acc(%)	Std	Acc(%)	Std	Acc(%)	Std
SVM	99.75	0.07	99.06	0.5	99.69	0.06	99.23	0.32
MLP	99.41	0.14	98.95	0.36	99.41	0.19	98.53	0.52
OPF	-	-	99.09	0.32	-	-	98.85	0.37
KNN	-	-	99.03	0.35	-	-	98.52	0.43

(a) Average global 7 classes accuracy

	FS13				FS20			
	Training		Testing		Training		Testing	
	Acc(%)	Std	Acc(%)	Std	Acc(%)	Std	Acc(%)	Std
SVM	100	0	99.94	0.09	100	0	99.93	0.11
MLP	99.92	0.07	99.8	0.22	99.98	0.02	99.92	0.09
OPF	-	-	99.87	0.11	-	-	100	0
KNN	-	-	99.9	0.1	-	-	99.99	0.04

(b) Average binarized accuracy

often with the same amount of shorted turns (HI-1/LI-1). Considering only the detection of failures, FS13 + SVM presented a very similar result, with 99.93 % of defects being detected and also keeping 0 % of FP rate. The difference is a higher confusion between failure types and that a few LI-1 patterns are also misclassified as normal, what is worse than confusing HI-2 patters with normal. Those points can be observed in the confusion matrix of Table 31.

The best detection of failures in general is achieved by FS20 and OPF, with 100 % accuracy considering only healthy and defective classes and $98.85\% \pm 0.37\%$ for 7 classes. It also presented good distinction of failure types, as shown in the confusion matrix of

Table 29 – Test set binary confusion matrices for all best combinations of features set and classifiers

SVM			MLP		
Real/Predicted	Normal	Failure	Real/Predicted	Normal	Failure
Normal	100	0	Normal	99.21	0.7857
Failure	0.0668	99.93	Failure	0.1119	99.89

KNN			OPF		
Real/Predicted	Normal	Failure	Real/Predicted	Normal	Failure
Normal	100	0	Normal	100	0
Failure	0.1111	99.89	Failure	0.145	99.86

(a) FS13

SVM			MLP		
Real/Predicted	Normal	Failure	Real/Predicted	Normal	Failure
Normal	100	0	Normal	99.57	0.4286
Failure	0.07803	99.92	Failure	0.0224	99.97

KNN			OPF		
Real/Predicted	Normal	Failure	Real/Predicted	Normal	Failure
Normal	99.93	0.07087	Normal	100	0
Failure	0	100	Failure	0	100

(b) FS20

Figure 32. FS13 + OPF achieved slightly worse detection of failures, with FN = 0.145 %, but better global accuracy, $99.09\% \pm 0.32$, than FS20+OPF. This and the results presented by MLP and KNN indicate FS20 represents better the differences of healthy and defective patterns, while FS13, between the types of failures.

The second best distinction between normal and failure patterns happens to FS20 + KNN, 99.99 % accuracy, with FN=0 % and FP=0.07 %. This result is so close to OPFs that it is considered equivalent. However, considering the distinction of failures, the differences are more significant. Table 33 presents the confusion matrix for FS20 + KNN. FS13 + KNN presented 99.9 % normal/failure accuracy with no FP cases.

MLP presented slightly worse results, despite also being very good. The combination of FS20 + MLP achieved very low FN = 0.0224 %. Even lower than SVM, with 0.07 %. However, the FP = 0.43 % still high compared to the other classifiers. Table 34 shows the confusion matrix for FS20 + MLP. The FP cases are caused by the misclassification of normal patterns as HI-2 and LI-1 patterns, while the FN, of HI-3 as Normal ones. The confusion between high impedance failures and normal patterns is more acceptable than low impedance and normal, considering the severity of the failure and the imminence of more severe damage.

Table 30 – Test set confusion matrix for the combination FS20 + SVM

Real/Predicted	Normal	HI-1	HI-2	HI-3	LI-1	LI-2	LI-3
Normal	100	0	0	0	0	0	0
HI-1	0	96.76	0	0	3.241	0	0
HI-2	0.4575	0	99.35	0	0	0.1961	0
HI-3	0	0	0.06481	99.94	0	0	0
LI-1	0	1.393	0	0	98.61	0	0
LI-2	0	0	0	0	0	100	0
LI-3	0	0	0	0	0	0	100

Table 31 – Test set confusion matrix for the combination FS13 + SVM

Real/Predicted	Normal	HI-1	HI-2	HI-3	LI-1	LI-2	LI-3
Normal	100	0	0	0	0	0	0
HI-1	0	97.67	0.1889	0	2.141	0	0
HI-2	0.2621	0.5898	97.9	0.2621	0.6553	0.3277	0
HI-3	0	0	0	100	0	0	0
LI-1	0.1295	1.425	0.2591	0	98.19	0	0
LI-2	0	0	0	0	0	100	0
LI-3	0	0	0	0	0	0	100

Table 32 – Test set confusion matrix for the combination FS20 + OPF

Real/Predicted	Normal	HI-1	HI-2	HI-3	LI-1	LI-2	LI-3
Normal	100	0	0	0	0	0	0
HI-1	0	96.59	0	0	3.41	0	0
HI-2	0	0	99.322	0	0.068	0.610	0
HI-3	0	0.128	0	99.680	0.192	0	0
LI-1	0	3.271	0.067	0	96.662	0	0
LI-2	0	0	0	0	0	100	0
LI-3	0	0	0	0	0	0	100

To summarize, FS20 seems to represent better the differences of healthy and defective patterns, while FS13, between the types of failures. Considering only normal and failure classes, the four classifiers presented very close results with FS20, all above 99.92 % of accuracy. However, OPF highlights for achieving a perfect distinction between normal and failure, followed by KNN, with only 0.07 % of FP. Meanwhile, SVM stands out with better global accuracy, keeping FP=0 % and almost insignificant FN, about 0.08 %.

To evaluate if FS20 is non linearly separable, the perceptron was applied to FS20. Compared to the results obtained in the preliminary investigation with perceptron in section 4.2.1, the accuracy increased significantly, from about $29.82\% \pm 3.76$ to 47.26 ± 2.96 . However, it still very low compared to nonlinear methods.

Table 33 – Test set confusion matrix for the combination FS20 + KNN

Real/Predicted	Normal	HI-1	HI-2	HI-3	LI-1	LI-2	LI-3
Normal	99.93	0	0.07087	0	0	0	0
HI-1	0	95.99	0	0	4.01	0	0
HI-2	0	0.3316	98.41	0	0.7294	0.5305	0
HI-3	0	0	0.1978	99.8	0	0	0
LI-1	0	3.64	0.5295	0	95.83	0	0
LI-2	0	0	0	0	0	100	0
LI-3	0	0	0	0	0	0	100

Table 34 – Test set confusion matrix for the combination FS20 + MLP

Real/Predicted	Normal	HI-1	HI-2	HI-3	LI-1	LI-2	LI-3
Normal	99.57	0	0.1428	0	0.2857	0	0
HI-1	0	95.26	0	0	4.7368	0	0
HI-2	0	0.0658	99.28	0.1974	0.1974	0.2632	0
HI-3	0.1316	0	0.1316	99.74	0	0	0
LI-1	0	3.2667	0.6	0	96.13	0	0
LI-2	0	0	0.0694	0	0	99.93	0
LI-3	0	0	0	0	0	0	100

5.4 Summary of features set and classifiers exploration

Consider FS0 as the feature set with calculations performed directly from the time domain signal (without using the wavelet transform) composed of skewness, kurtosis, mean absolute deviation, median absolute deviation, and frequency. FS4, FS13 and FS20 were already described throughout this chapter. Table 35 summarized the results for those features set and the 4 classifiers analyzed. SVM presented the best results for all those features set, combining the distinction between normal and failure patterns and also the separation of defect subtypes, simultaneously. The exception is for FS20, which OPF and KNN achieved higher binary accuracy than SVM. The results obtained with the DWT included are all superior to the features extracted from the time domain signal directly.

5.5 Conclusion

In this chapter, the inclusion of the Multiresolution Analysis, using the DWT, before calculating the statistical features was investigated. The number of levels of decomposition investigated were in the range from 2 to 6. ANOVA and F-score were used to eliminate some features that were disturbing the classification. The results were satisfactory.

The decomposition of the original flux signal in frequency bands of different resolution enables an increase in the dimensionality of features space since each feature can be calculated separately for each frequency band. It also allows the removal of frequency

Table 35 – Summary of features and classifiers results considering 7 classes

Features set	Classifier	7 classes		7classes - Binarized			
		Acc (%)	Std	TN (%)	TP (%)	Acc (%)	Std
FS0	SVM	96.25	0.75	98.65	99.47	99.35	0.35
	OPF	95.52	0.71	96.83	99.57	99.21	0.32
	KNN	94.69	1.14	97.39	99.51	99.23	0.35
	MLP	94.2	1.36	97.79	99.44	99.22	0.46
FS04	SVM	98.59	0.48	99.72	99.85	99.84	0.15
	OPF	97.87	0.9	98.5	99.87	99.68	0.28
	KNN	97.26	0.55	98.52	99.71	99.54	0.24
	MLP	97.83	0.76	98.14	99.83	99.6	0.25
FS13	SVM	99.06	0.5	100	99.93	99.94	0.09
	OPF	99.09	0.32	100	99.86	99.87	0.11
	KNN	99.03	0.35	100	99.89	99.9	0.1
	MLP	98.95	0.36	99.21	99.89	99.8	0.22
FS20	SVM	99.23	0.32	100	99.92	99.93	0.11
	OPF	98.85	0.37	100	100	100	0
	KNN	98.52	0.43	99.93	100	99.99	0.004
	MLP	98.53	0.52	99.57	99.97	99.92	0.09

bands coefficients that are not related to the problem, like D_1 and D_2 . Those aspects combined contributed to a better representation of the problem, improving the classification for all the analyzed classifiers.

The classification global accuracy improved from about 95 % to about 99 % with FS13 and FS20. The true positive rate from about 99.5 % to 99.9 %. Binary classification achieved about 99.9 %. The analyzed classifiers, SVM, OPF, MLP, and KNN, presented results close to each other. However, SVM in general achieves slightly better results in all the metrics simultaneously, while OPF stands out in combination with FS20, achieving 100 % binary accuracy.

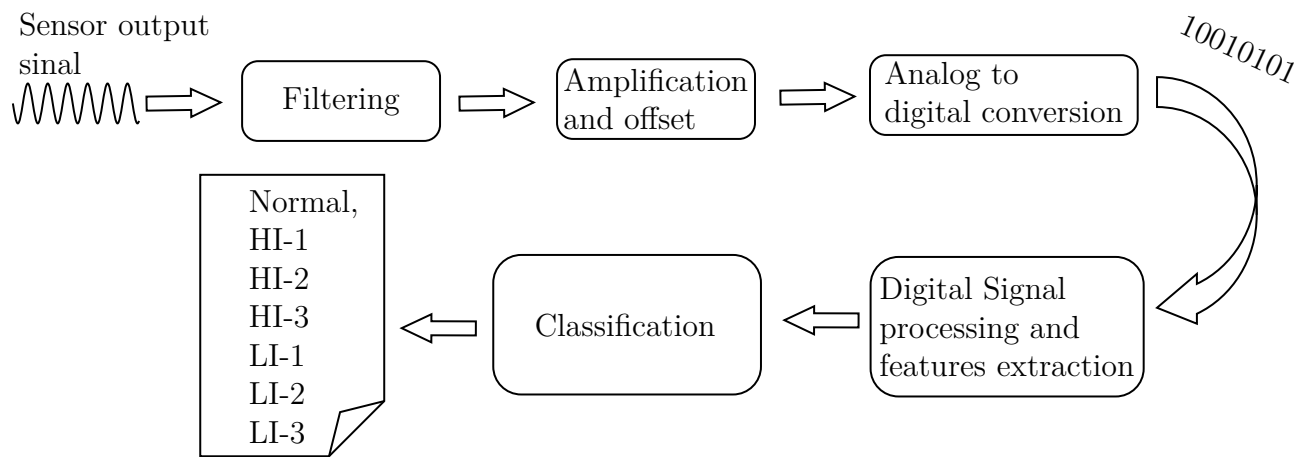
From the embedded design point-of-view, KNN and OPF classifiers have the disadvantages of using the complete training set during prediction (target platform). Despite the possibility of storing this data in program memory, which is usually easily available in modern microcontrollers, to access them and perform all the calculations using this huge amount of data (2072 patterns with 14 features each for FS20), it would consume considerable execution time and energy. MLP is usually very cost effective when considering the memory footprint and execution time and, for this reason, it is usually chosen when considering embedded applications. However, with the microcontrollers cost reduction and improvements in memory and performance, it was decided to prioritize the better results obtained with SVM and consider it the first choice for this application. This way, SVM is the only classifier to be considered in the next experiments.

With the classifier defined, the next step is to choose between the feature sets

FS04, FS13 and FS20. In this analysis, the trade-off between classification performance and computational cost is the main factor to guide this project decision. The methodology used to make this choice is explained in the next chapter.

The diagram of Figure 34 describes the general idea of the proposed system. The flux signal is analogically filtered, amplified and offset to adequate to the ADC requirements. After it has been captured, the DWT is applied N times, to perform the decomposition in N resolution levels. The features are calculated and the SVM classifier make a prediction about the motor actual condition.

Figure 34 – Short-circuit failure detection proposed approach



Source: Author

6 Embedded prototyping

In this chapter it is described the process to reduce the computational resources usage and the initial stage of the embedded prototyping implementation, with metrics to evaluate it.

6.1 Computational resource usage reduction

The following approaches are analyzed with the objective of reducing the computational resources usage in the embedded prototyping:

- Group all failures and consider only normal and failure classes for the classification task;
- Total acquisition time (TAT) reduction;
- Sampling frequency (f_s) reduction;

6.1.1 2 classes approach

Despite it is interesting to know more information about the type of failure, the main objective is to detect it as soon as possible. Moreover, using low-cost equipment to make this solution practicable even for small size induction motors. This way, using a binary approach might be a good solution to save computational resources in the classification phase.

In the previous chapter, SVM was defined the best classifier considering classification performance and the drawbacks of KNN and OPF. In the Table 36 there are the classification results obtained for classifiers trained using the normal and failure labels only. When all the failures are grouped, there are 7 times more failure than normal patterns. To balance the dataset, the number of failure patterns was reduced uniformly, which resulted in 800 patterns, where 640 (80 %) were used for training the SVM classifier. The results were worse than considering the 7 classes. Because the number of failure patterns was significantly reduced, there are less defective examples to be used during the classifier training phase. Consequently, it does not learn to identify failure patterns as well as with the 7 classes approach. It learns well to identify the normal ones, achieving 100 % true negative rate, meanwhile lower true positive rate. The average confusion matrix for the best feature set with SVM is displayed on Table 37.

Table 36 – Summary of preliminary study considering only normal and failure labels for classification

	Binary				7 classes binarized			
	Training		Testing		Training		Testing	
	Acc(%)	Std	Acc(%)	Std	Acc(%)	Std	Acc(%)	Std
FS0	99.15	0.26	97.13	1.32	100	0	99.35	0.35
FS4	99.92	0.07	98.66	1.04	100	0	99.84	0.15
FS13	99.99	0.03	99.62	0.41	100	0	99.94	0.09
FS20	100	0	99.69	0.42	100	0	99.93	0.11

Table 37 – Test set average confusion matrix for FS20+SVM

Real/Predicted	Normal	HI
Normal	100	0
HI	0.5637	99.44

This brief experiment intended only to evaluate if a two class approach was a viable option to optimize computational resources usage in classification stage and more experiments are necessary to confirm the hypothesis that using a binary approach is possible to achieve results as good as using all the seven types. For example, increasing the normal patterns instead of decreasing so much the failure examples. This preliminary study presented encouraging accuracy results and decreased significantly the number of support vectors, because instead of having 21 binary classifiers, in a one-versus-one configuration, only one binary classifier apply.

Check Table 38 to compare the number of support vectors between feature sets and between 7 and 2 classes approaches. The combination of FS20 and binary classification (90.2 ± 3.79 SV) represents a number of support vectors 85 % lower than FS20 + 7 classes (610.85 ± 7.91 SV) classification. However, if instead of FS20, FS13 is chosen, even considering the 7 classes approach (367.25 ± 8.5 SV), the SV number is about 40 % lower and the binarized accuracy ($99.94\% \pm 0.09$) still better than FS20 in 2 class approach $99.93\% \pm 0.11$. Thus, the execution time and memory footprint of the feature extraction have an important impact on this decision, establishing priorities and determining how much it is necessary to compromise the classifier. With that information, it is easier to evaluate the benefits of the binary approach.

This way, in the following analysis, 7 classes are considered and the binary classification approach is left as an option for further reductions if necessary, or an idea for more constrained implementations. After analyzing the impact of each stage of the processing in response time and memory footprint on the target embedded platform, this option should be reanalyzed and the decision can be more accurate and justified.

Table 38 – Comparison of support vectors number

Features set	Number of support vectors			
	7 classes		2 classes balanced	
	mean	std	mean	std
FS0 (5 features)	602.6	7.77	70.05	3.6
FS4 (6 features)	479.45	7.08	62.45	3.02
FS13 (7 features)	367.25	8.5	108.75	2.54
FS20 (14 features)	610.85	7.91	90.2	3.79

6.1.2 Total Acquisition time reduction

All the study described until this point considered a total acquisition time of 10s. This way, even if the time necessary to process and classify the pattern was negligible, only after 10 seconds any conclusion about the stator winding situation would be provided. The elapsed time between the starting of a short circuit and a development of a severe failure condition, that would seriously damage the motor, is very hard to precise, because it can range from a few seconds to hours. Anyway, the shorter the interval between analyzes, the better the failure evolution is tracked and the greater the chances the machine will have its operation interrupted before a significant evolution of the defect. This way, it is interesting to investigate how much the total acquisition time can be reduced without significantly compromising the results achieved.

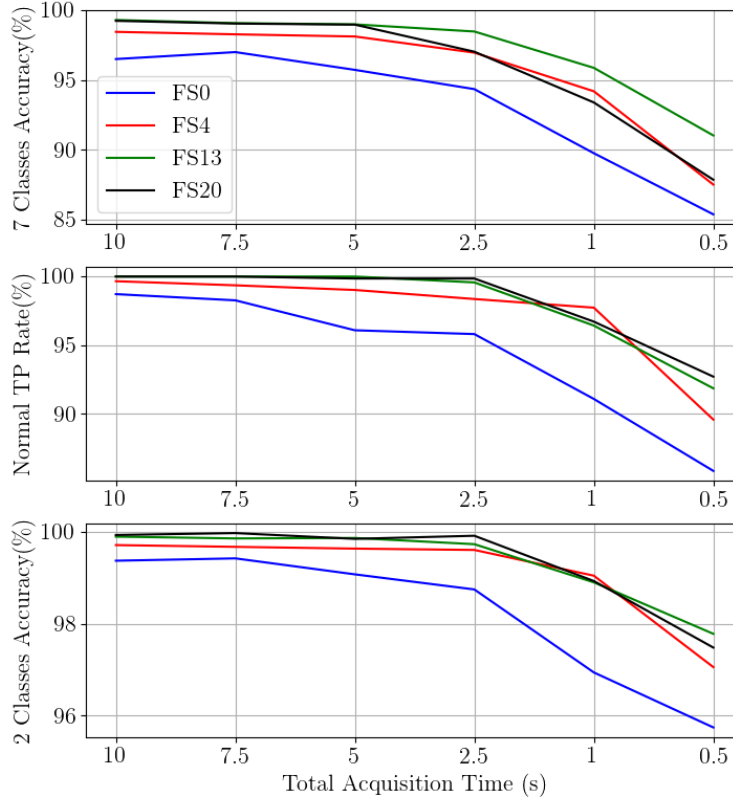
The original combination of sampling frequency and total acquisition time, respectively 10 kHz and 10 s, results in 100000 points ($N_p = TAT \times f_s$). This way, this analysis is very important to define the signal acquisition parameters for the embedded application, because 100,000 points take a huge memory space and would also imply in significant execution time during the features extraction phase.

The DWT is applied to FS4, FS13 and FS20 in different number of decomposition levels and at each level, a down-sampling operation happens. This way, the reduction in total acquisition time and, consequently, in the number of acquired points, can impact with different intensity in each feature set's abilities to represent the problem. Taking this into account, despite FS20 presented the best results for $TAT = 10$ s, it is important to evaluate and compare the performance of the feature sets for smaller TAT.

To perform this analysis, the same dataset patterns used in previous studies were truncated to compose other datasets with smaller TAT. The values chosen were 7.5 s, 5 s, 2.5 s, 1 s and 0.5 s. The same procedures of the previous studies were reproduced: grid search with k-fold cross-validation to find SVM parameters σ and C , followed by 20 training/testing independent experiments. The same amount of patterns was used for training (80 %) and testing (20 %).

Figure 35 presents the evolution of the 7 class average accuracy, normal TP rate

Figure 35 – Evolution of the classification performance for different values of total acquisition time (TAT)



and normal/failure classification accuracy for the 4 different feature set analyzed, when the total acquisition time decreases from 10 s to 0.5 s, using SVM. Analyzing the graphics, it is clear that using the wavelet transform improves significantly the results, because even when the total execution time is reduced 20 times (0.5 s), the accuracy results and normal TP rates are higher for features set 4, 13 and 20.

FS13 and FS20 presented very similar results for $TAT \geq 5$ s. When the total execution time is decreased further, FS20's 7 classes accuracy degrades more significantly than the other features sets results. It was already expected that the FS which the DWT was performed in more levels would have its performance degraded more significantly by the decrease in TAT. Because it corresponds to a decrease in the number of points in the original time domain signal and, at each level of the decomposition, the signal is down-sampled. This way, for more levels of decomposition, the impact of the TAT reduction is more pronounced. Table 39 describes the number of points in the last level of the decomposition of each combination of feature set and TAT. However, the Normal TP rate and binary accuracy results are still very similar for FS20 and FS13, considering $TAT \geq 5$ s.

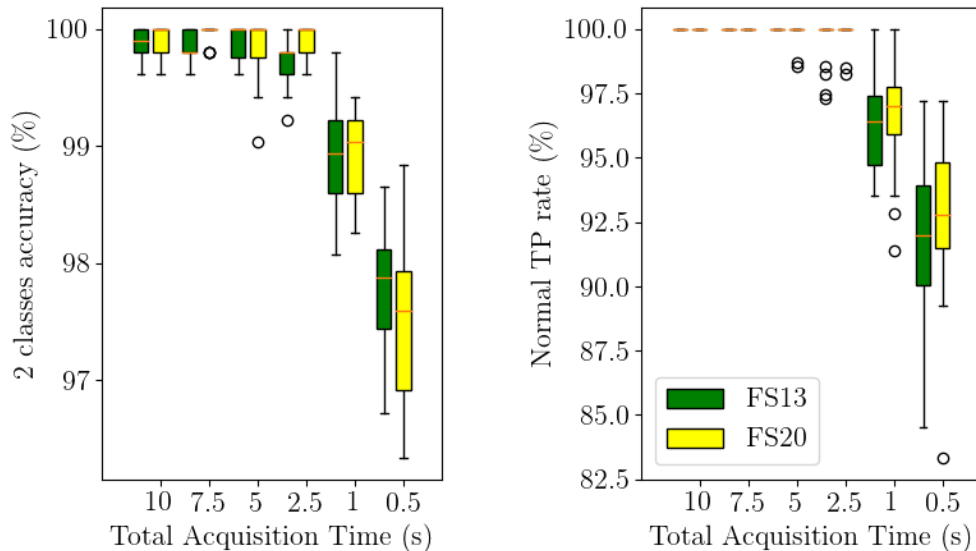
Table 39 – Number of points for the last level of decomposition for each combination of TAT and FS

TAT (s)	10	7.5	5	2.5	1	0.5
N° of points	100,000	75,000	50,000	25,000	10,000	5,000
Features set	N° of points at the last level of decomposition					
FS4 (2 levels)	25,000	18,750	12,500	6,250	2,500	1,250
FS13 (3 levels)	12,500	9,375	6,250	3,125	1,250	625
FS20 (5 levels)	3,125	2,343	1,562	780	312	156

As highlighted before, the most important aspect of this investigation is to detect the failure while not creating false alarms. Taking this into account, the box-plot of 2 classes accuracy and normal TP rate for 20 independent experiments of training/testing for the best two features set, FS13 and FS20, are plotted in Figure 36.

The box-plots of Figure 36 confirm what is indicated in Figure 35: for $TAT \geq 2.5$ s, there is not a huge impact on binary accuracy and normal TP rate. The results for both feature sets are very similar, but FS20 seems slightly better. A reduction from 10 s to 2.5 s, using FS20, is very cost effective, because with a quarter of the total acquisition time, which implies 75 % less data points to be stored and processed and much faster predictions, it is possible to maintain a result quite close to the original configuration. The median value for both metrics still 100 %, what means that at least for 50 % of the independent experiments (10), the binary accuracy and normal TP rate were 100 %. Even in the worst execution, the binary accuracy still above 99.5 %. From the 20 independent experiments, only two of them got Normal TP lower than 100 %, characterized as outliers.

Figure 36 – Normal/failure accuracy and normal TP rate box-plot comparison for FS13 and FS20 over total acquisition time reduction



However, Figure 35 indicates a loss in the capacity to distinguish among different types of failures, represented by the decrease in global accuracy from 99.23 % to 97 %.

6.1.3 Sampling frequency reduction

A study on the impact of reducing the sampling frequency (f_s) in classification performance was also carried out. The same patterns used in previous experiments were digitally filtered, to avoid aliasing error, and downsampled to compose new datasets with $f_s = [5, 3.3, 2.5, 2, 1.25, 1]$ kHz. The same process to extract the features, train and predict was performed. The digital filter was 5th order Butterworth, with cutoff frequency $f_c = \frac{f_s}{2} - 300$.

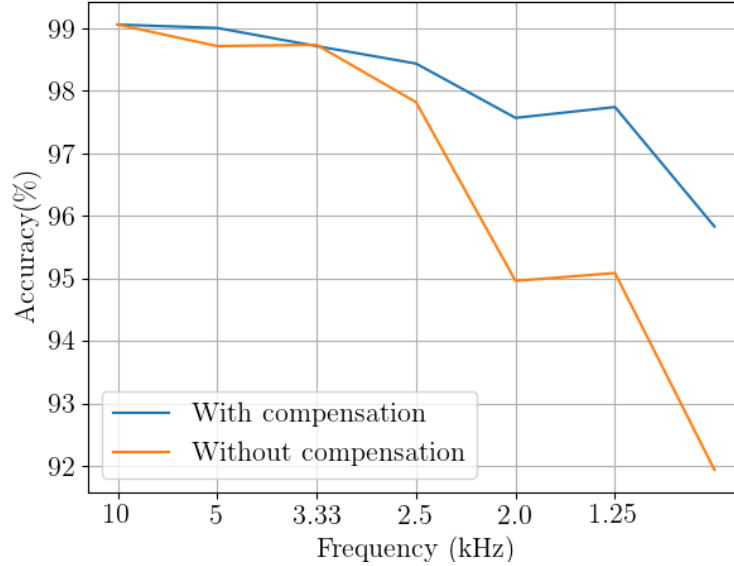
The way the Wavelet Transform and MRA (Multiresolution Analysis) works, the frequency band comprised on the approximation and detail coefficients of each level depends on the Nyquist frequency of the original signal, as described in Tables 16 and 24. This way, when reducing the sampling frequency of the signal, it does not make sense to keep the same number of decomposition levels. For instance, if $f_s = 5$ kHz, then only one level of decomposition should be performed to get the frequency components of sub bands 0 to 1.25 kHz and 1.25 kHz to 2.5 kHz, that were originally obtained with 2 levels of decomposition. Therefore, for FS13, that uses 3 levels of decomposition, the compensation should happen as described in Table 40. Each time f_s is halved, one level is reduced, except for $f_s = 1.25$ kHz, because it is the first level already. To show the importance of this compensation, on Figure 37, the accuracy for FS13 and the reduced sampling frequencies with and without the described compensation are plotted. Notice the accuracy decays more smoothly if the levels are compensated than if the original number of levels is maintained.

In Figure 38 the 7 classes accuracy, normal TP rate and 2 classes accuracy, for 20

Table 40 – Decomposition levels adjusted to f_s reduction for FS13

$f_s(kHz)$	N. levels	Features
10	3	Mean Absolute Deviation A3, Median Absolute Deviation A3 and D3, Kurt A3, Ep A3 and D3, Frequency
5	2	Mean Absolute Deviation A2, Median Absolute Deviation A2 and D2, Kurt A2, Ep A2 and D2, Frequency
3.3	2	Mean Absolute Deviation A2, Median Absolute Deviation A2 and D2, Kurt A2, Ep A2 and D2, Frequency
2.5	1	Mean Absolute Deviation A1, Median Absolute Deviation A1 and D1, Kurt A1, Ep A1 and D1, Frequency
2	1	Mean Absolute Deviation A1, Median Absolute Deviation A1 and D1, Kurt A1, Ep A1 and D1, Frequency
1.25	1	Mean Absolute Deviation A1, Median Absolute Deviation A1 and D1, Kurt A1, Ep A1 and D1, Frequency
1	1	Mean Absolute Deviation A1, Median Absolute Deviation A1 and D1, Kurt A1, Ep A1 and D1, Frequency

Figure 37 – Accuracy comparison for FS13, with and without the compensation in the number of levels depending on f_s



independent experiments, are presented for the different f_s described. The experiment showed that the usage of Wavelet Transform is a better approach than calculating the features directly from the original signal, for $f_s \geq 2$ kHz. However, for smaller f_s values, it depends on the metric and the feature set. FS20 was the feature set which presented the smoothest loss in evaluated metrics when f_s was reduced.

Similar to the TAT analysis, In Figure 39 there are the box-plots of binary accuracy and normal TP rate using FS20, over the frequency reduction analysis. It can be reduced from 10 kHz to 2.5 KHz without significant losses in classification results. The binary accuracy even improves when f_s is reduced from 10 kHz to 2.5 kHz frequency. Considering that from 20 independent experiments, only two did not present 100 % Normal TP rate.

In the next section, it is described the study to combine a reduction in sampling frequency and in total acquisition time.

6.1.4 Total acquisition time and sampling frequency reduction combined

After verifying it is possible to reduce the sampling frequency from 10 kHz to 2.5 kHz or the total acquisition time from 10 s to 2.5 s, with FS20 without significant loss in failure detection capacity, a similar study combining the reduction in both parameters was carried out. The grid analyzed was composed by the combination of $Fs = [10, 5, 3.3, 2.5]$ kHz and $TAT = [10, 7.5, 5, 4.5, 4, 3.5, 3, 2.5]$ s. The same procedures applied throughout the classification experiments were performed. The classifier used was SVM and the same number of patterns were applied for training and testing stages.

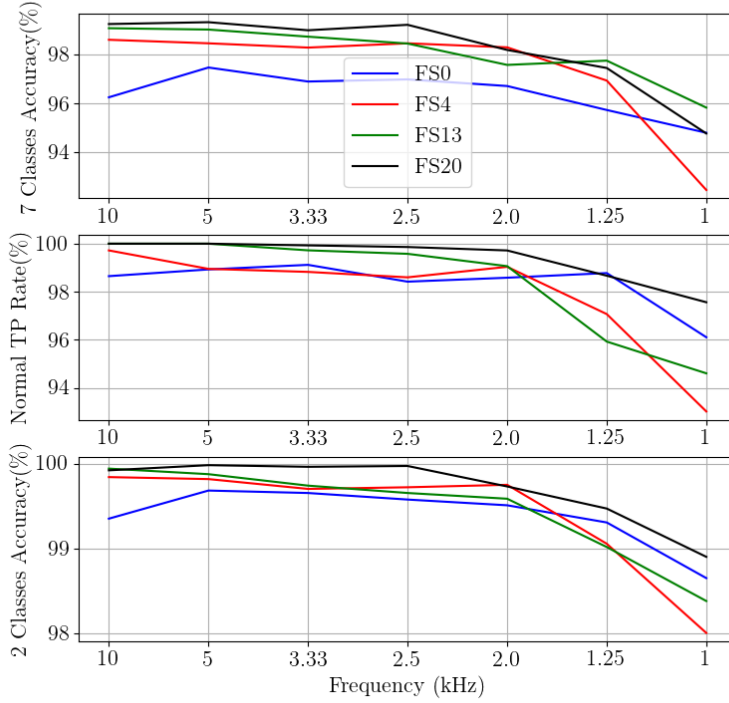
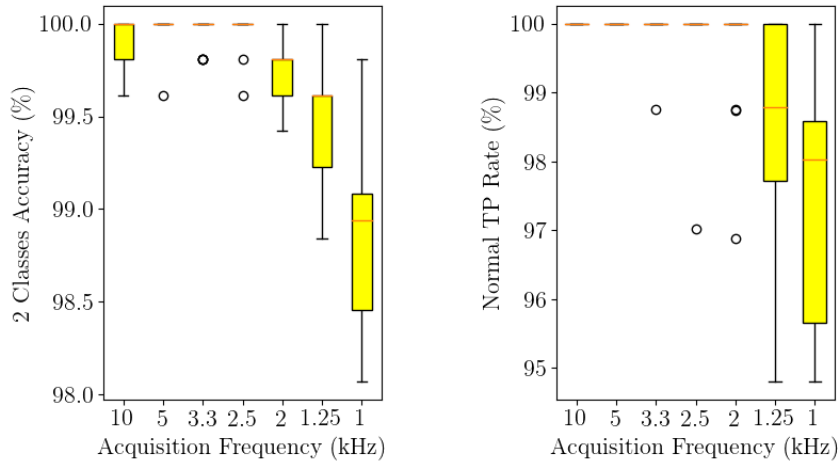
Figure 38 – Average results for different values of f_s Figure 39 – Normal/failure accuracy and normal TP rate box-plot for FS20 over f_s reduction

Figure 40 contain tables with the average accuracy for 7 classes (40a), Normal TP rate (40b), average 2 classes accuracy (40c) and the number of points for each combination of TAT and f_s (40d). The definition of the ideal configuration depends on some aspects: it should try to balance accuracy, specially 2 classes, normal TP rate and total number of points. Also, between f_s and TAT, the priority is to reduce the second, because, as already highlighted, the short-circuit defect has a fast evolution, sometimes a few seconds.

The results showed its possible to reduce both the parameters and still achieve very good results, with 7 classes accuracy above 96 %, Normal TP rate ≥ 98.75 % and 2 classes accuracy ≥ 99.72 %. The decision should be taken after analyzing the embedded platform resources and after a preliminary study of execution time and memory footprint of the proposed short-circuit detection method on the embedded platform. This is described in the next section.

6.2 Embedded application platform and model choice

The application described in this dissertation is computationally intense, because it should process a huge amount of data using complex signal processing and machine learning algorithms. Furthermore, the response time should be very small, because the failure can drastically evolve in only a few seconds, in some conditions, causing severe damage to the machine. Based on those characteristics, a 32-bit microprocessor with DSP instructions set and a floating point unit seems a good option. Based on the previous section, the data to be processed is an array of size in the range from 6,250 to 100,000 points, depending on total acquisition time and sampling frequency combination. Thus, RAM memory capacity is also another key point on platform choice.

With these characteristics the STM32F407VGT6 was chosen. STM32F407xx family is based on the ARM Cortex-M4, a 32-bits RISC core which presents good processing performance combined with low power consumption. It is designed with a 3-stage pipeline Harvard architecture, making it ideal for demanding embedded applications. It also has hardware division, fast digital-signal-processing oriented multiply accumulate arithmetic and SIMD operations for signal processing applications. It also has an IEEE754-compliant single-precision floating-point unit (ARM, 2011).

The STM32F407VGT6 operates at a clock frequency of up to 168 MHz, incorporates high-speed embedded memories of 1 Mbyte Flash, 192 KBytes of RAM and 12-bit ADC. It also has an ART Accelerator, which is a memory accelerator optimized for STM32 industry-standard ARM Cortex-M4 with FPU processors, which prevents the processor to wait for the Flash memory at higher frequencies (STMICROELECTRONICS, 2016).

Based on the microcontroller specification and on the embedded implementation of the DWT by the author (CUNHA et al., 2018), an estimate of the maximum number of points can be performed to help defining f_s , TAT and finish the model definition. This DWT embedded implementation is part of another case of study using the same methodology for a similar embedded application including signal processing and machine learning. The problem addressed is failure detection of energized polymeric insulators and it will be described later in this chapter.

Extrapolating the DWT implementation memory footprint analysis (CUNHA

Figure 40 – Results for simultaneous reduction in TAT and f_s using FS20 and SVM

	10 s	7.5 s	5 s	4.5 s	4 s	3.5 s	3 s	2.5 s
10 kHz	99.23	99.03	98.95	98.34	98.17	97.18	97.4	97.02
5 kHz	99.31	99.2	98.64	98.56	98.27	98.07	97.26	96.23
3.33 kHz	98.98	98.6	98.18	97.76	97.81	97.89	97.49	96.73
2.5 kHz	99.19	99.09	98.35	98.15	98.15	98.05	97.72	96.94

(a) Average 7 classes accuracy

	10 s	7.5 s	5 s	4.5 s	4 s	3.5 s	3 s	2.5 s
10 kHz	100.0	100.0	99.86	99.79	99.71	99.29	99.86	99.85
5 kHz	100.0	100.0	99.93	99.79	99.86	99.63	99.79	99.72
3.33 kHz	99.93	99.93	99.93	99.19	99.42	99.57	99.12	98.75
2.5 kHz	99.86	100.0	99.42	99.71	99.71	99.5	99.64	99.5

(b) Average Normal TP rate

	10 s	7.5 s	5 s	4.5 s	4 s	3.5 s	3 s	2.5 s
10 kHz	99.93	99.97	99.85	99.88	99.86	99.81	99.83	99.91
5 kHz	99.98	99.94	99.98	99.88	99.9	99.91	99.92	99.86
3.33 kHz	99.96	99.99	99.95	99.8	99.85	99.91	99.8	99.72
2.5 kHz	99.97	99.97	99.81	99.86	99.91	99.83	99.8	99.72

(c) Average 2 classes accuracy

	10 s	7.5 s	5 s	4.5 s	4 s	3.5 s	3 s	2.5 s
10 kHz	100000	75000	50000	45000	40000	35000	30000	25000
5 kHz	50000	37500	25000	22500	20000	17500	15000	12500
3.33 kHz	33330	24997	16665	14998	13332	11665	9999	8332
2.5 kHz	25000	18750	12500	11250	10000	8750	7500	6250

(d) Number of points

Source: Author

et al., 2018), performed for float32 data type, an input data array of up to 20,000 points might fit the chosen device memory and also leave enough space for classifier implementation. Analyzing the metrics of the options presented in the previous section that fulfill these requirements, the 3 s total acquisition time and 5 kHz sampling rate was chosen, because it presents the best cost-benefit considering the 7 classes, binary accuracy, normal class sensitivity and lower TAT. The average confusion matrix for 20 training/testing independent experiments after defining the SVM parameters σ and C using a grid search for this configuration is presented in Table 41.

Comparing this model to the original one, before the reduction in the number of points from 100,000 to 15,000, there is a small decrease in the model 7 classes mean accuracy from 99.23 % to 97.26 %, but it still a very good result, considering the true positive rate of all the classes are higher than 91.5 %. Moreover, the false alarm rate (FP)

Table 41 – Average confusion matrix for Feature set 20 +SVM, with $f_s = 5$ kHz, $TAT = 3$ s $C = 10000$ and $\sigma = 0.25$

Real/Predicted	Normal	HI-1	HI-2	HI-3	LI-1	LI-2	LI-3
Normal	99.78	0	0.2133	0	0	0	0
HI-1	0	94.106	0.3311	0	5.56	0	0
HI-2	0.1294	0.8409	96.25	0.3881	2.2	0.1940	0
HI-3	0.1296	0	0.1296	99.68	0.0648	0	0
LI-1	0	6.739	1.476	0	91.78	0	0
LI-2	0.0722	0	0	0	0	99.93	0
LI-3	0	0	0	0	0	0	100

is only 0.2 % of the normal patterns and the binary mean accuracy is 99.92 %. From the 20 models, the one which presented the best combination of binary or 7 classes accuracy and normal class sensitivity was chosen to be embedded. Table 42 summarizes the parameters of the proposed solution.

Table 42 – Proposed solution Parameters

Parameter	Value
Data acquisition	
F_s	5 kHz
Total acquisition time	3 s
Number of points	15,000
Pre-processing	
Wavelet Mother	Daubechies 2
Number of decomposition levels	4
Number of features	14
Classification	
Classifier	SVM
Kernel	RBF
σ	0.25
C	10,000
Number of support vectors	720
Multi-class approach	OVO, 21 binary classifiers
Number of support vectors per class	[108,132,130,114,125,60,51]
Training set percentage	80 %

6.3 Embedded application implementation

The pre-processing and classification stages of the stator winding short-circuit detection system proposal, described in the previous section, which the parameters are summarized in Table 42, were implemented on the embedded platform.

An embedded DWT library was implemented using functions of the CMSIS library,

in order to be generic enough to be used in very different applications and with any processor of the ARM Cortex-M family. It works as a hardware abstraction layer between the processor and the peripherals (STMICROELECTRONICS, 2018). The DWT library was built using filter bank, convolution, and subsampling operations as basic functions. This approach was chosen instead of the lifting scheme, based on the results obtained by Ukil and Barlocher (2007), where the classical algorithm was 3 times faster and occupied less program memory.

Two versions of the DWT were implemented: float32, a 32-bit floating point data type, and Q15, 16-bit fixed point type. However, the execution time comparison between float32 using the FPU and Q15 at filtering and FFT operation described in STMicroelectronics (2018) application note showed the difference is not very significant to justify the Q15 implementation, taking into account the precision loss and implementation difficulties to avoid overflow and compensate for the limited range of fixed-point data representation. Furthermore, the metrics of the Polymeric Insulation online detection (CUNHA et al., 2018) implemented by the author, which used the same DWT implementation, using both float32 and Q15 confirms those results. For more details about this application, check Appendix-A. The advantages of Q15 implementation are related to memory usage, considering it represents the data as a 16 bits number, while float32 uses 32 bits. This way, the analysis with Q15 of this application is not presented.

The statistical features Mean Absolute Deviation, Median Absolute Deviation, Skewness and Kurtosis were implemented using float32 data type. The SVM prediction was also implemented using float32 and the parameters are all stored in flash memory (non-volatile). The first step in SVM prediction is to calculate the kernel values for all the support vectors. After this, the discriminant function (equation 2.68) is calculated for each pair of classes in OVO (one-vs-one) multi-class approach. There is a vector *StartIdx* whose elements represent the start index of each class in the kernel vector. The *N_SVperClass* vector stores the number of support vectors in each class. Another array, *dual_coef* stores a matrix with the result of the product $y_i\alpha_i$ for each support vector and each classifier. The *Intercept* vector stores the bias for all the classifiers. The pseudo-code Algorithm 1 describes the SVM prediction implemented.

In the next section, the methodology and the results of the embedded prototype evaluation are discussed.

6.3.1 Embedded prototype evaluation

To validate the embedded implementation, from DWT to SVM prediction, a subset of the dataset composed of 519 patterns with examples of all the combinations of classes, load and driving frequency was serially sent from the working station to the microcontroller and the classification result sent back to the working station to be compared to the results

Algorithm 1 OVO (one-vs-one) SVM prediction

```

1: function KERNELRBF(InputPattern) ▷ Calculate kernels for all the support vectors
    $i = 0$ 
2:   while  $i < N\_SV$  do  $Kernel[i] = \exp(-\gamma ||\mathbf{x}_i - \mathbf{x}||^2)$ 
    $i++$ 
   return Kernel
3: function SVM PREDICT(Pattern)
   Kernel=KERNELRBF(Pattern)
    $i = 0$ 
4:   while  $i < N\_CLASSES$  do  $j = i + 1$ 
5:     while  $j < N\_CLASSES$  do ▷ Dot product between kernel of class i support
       vectors and DualCoef corresponding to classes i and j
       CoefInit=StartIdx[i]+(j-1)*N_SV
       CoefEnd=CoefInit+N_SVperClass[i]
       KernelInit = StartIdx[i]
       KernelEnd = KernelInit+N_SVperClass[i]
        $a = \text{dot\_product}(\text{DualCoef}[\text{CoefInit}...\text{CoefEnd}], \text{Kernel}[\text{KernelInit}...\text{KernelEnd}])$ 
       ▷ Dot product between kernel of class j support vectors and DualCoef corresponding
       to classes j and i
       CoefInit=StartIdx[j]+i*N_SV
       CoefEnd=CoefInit+N_SVperClass[j]
       KernelInit = StartIdx[j]
       KernelEnd = KernelInit+N_SVperClass[j]
        $b = \text{dot\_product}(\text{DualCoef}[\text{CoefInit}...\text{CoefEnd}], \text{Kernel}[\text{KernelInit}...\text{KernelEnd}])$ 
       ▷ Discriminant function
        $Out = a + b + \text{Intercept}[\text{counter}]$ 
6:       if  $Out > 0$  then  $\text{votes}[i]=\text{votes}[i]+1$            ▷ Add vote to the winning class
7:       else
        $\text{votes}[j]=\text{votes}[j]+1$ 
        $\text{counter}++$ 
        $j++$ 
        $i++$ 
       Prediction= $MaxIdx(\text{Votes})$ 
       return Prediction

```

obtained in computational simulation. The results were the same for all the 519 patterns validating the embedded version as equivalent to the algorithm developed in the working station in Python. In the next subsections, the execution time and memory footprint of the embedded prototype are evaluated.

6.3.1.1 Execution time

To evaluate the impact of the FPU (Floating Point Unit) usage, the execution time was measured with software and hardware implementation of floating point operations. The results are presented on Table 43. With the FPU enabled, the execution time decreases significantly, from 224 ms to 104 ms, approximately. The pie chart of Figure 41 illustrates the percentage contribution of each stage of the algorithm to the execution time, with the FPU enabled. For both scenarios, feature extraction (Wavelet Decomposition and Features Calculation) takes about 3/4 of the execution time, despite the SVM prediction comprises 21 binary classifiers and a total of 720 support vectors. It is important to highlight the feature extraction still taking the most significant part of the execution time, even after the efforts to decrease the number of points from 100,000 to 15,000, by means of f_s and TAT reduction, and also some feature elimination using ANOVA. If this optimization was not performed, this value would be extremely higher or even forbid the implementation in this platform due to the huge memory usage.

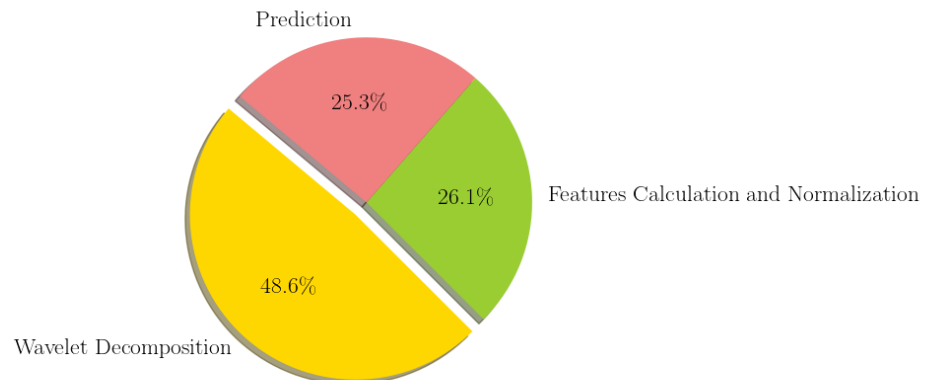
Table 43 – Execution time comparison with and without the FPU

Stage	Software implementation of floating point operations (μs)	Hardware implementation of floating point operations (μs)
Wavelet Decomposition	124560	49170
Features Calculation	60324	26391
Normalization	19	9
Prediction	39660	25604
Total	224563	104086

Related to the statistical features calculation, Median Absolute Deviation is the most time consuming, because it includes ordering the vector to obtain the median value. For the forth level approximation coefficients, A_4 , which is a vector of 938 elements, it takes 2560 μs , while Mean Absolute Deviation takes 706 μs , kurtosis 1505 μs and Skewness 1350 μs , with the FPU enabled. The relative energy calculation of all the decomposition levels takes 2912 μs . In Prediction, 24546 μs are spent to calculate the Kernel for all the 720 support vectors and only 1058 μs for the remaining computations.

Including the acquisition, the total response time (diagnostic time) is about 3.1 s (FPU enabled).

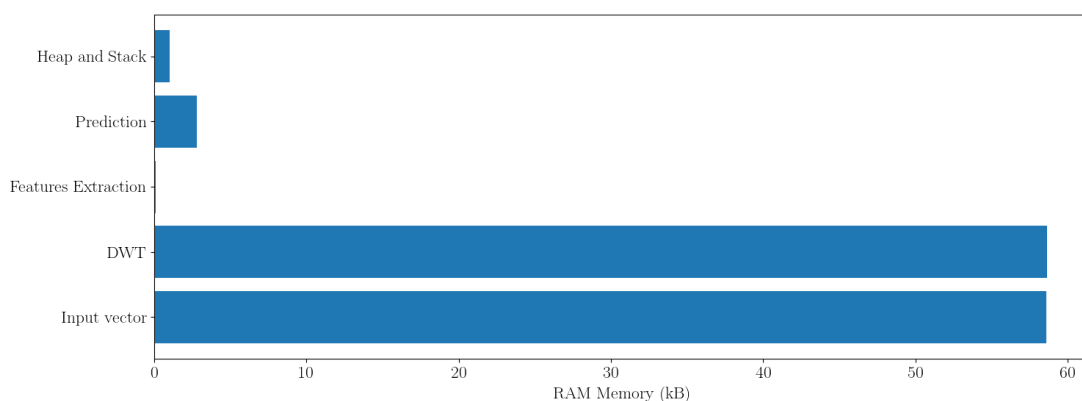
Figure 41 – Total execution time distribution with FPU enabled



6.3.1.2 RAM memory footprint

This application has intense RAM memory usage because of the input vector length, 15000 points, and the data type used, float32. Only this occupies 58.59 kB. The lifting scheme of the DWT does not allow in place computation because it would overwrite data that still necessary for calculation. Thus, the RAM memory used only for DWT is twice the input vector size. After the DWT is executed, the input vector is used as temporary variable for features calculation that requires it. Fortunately, the other functions do not require much RAM. The graphic of Figure 42 shows the RAM memory used in each stage of the algorithm. In total 122.5 kB were used and from this total, 96 % was used to store the input vector and the output of the wavelet decomposition. Using a smaller data type, like Q15, would significantly reduce the RAM memory usage, despite the loss in precision and increased effort to avoid overflow.

Figure 42 – RAM memory usage



6.3.1.3 FLASH memory footprint

In total, the application uses 71.74 kB of FLASH memory, in which, as exposed in the pie chart of Figure 43, only 18.9 % are occupied by code, about 80 % is used to store constant values and 0.5 % to Interrupt Service Routine. Analyzing in details the data storage distribution, displayed in Table 44, it is clear that, unlike execution time and RAM memory footprint, where features extraction takes the greatest amount, the prediction is in charge of the highest FLASH memory usage, with almost all the data stored in FLASH (not including code) being the SVM parameters, about 97.47 %. However, currently, high-performance microcontrollers as required for this application has a huge amount of FLASH memory and RAM is the most critical point.

Figure 43 – FLASH memory usage

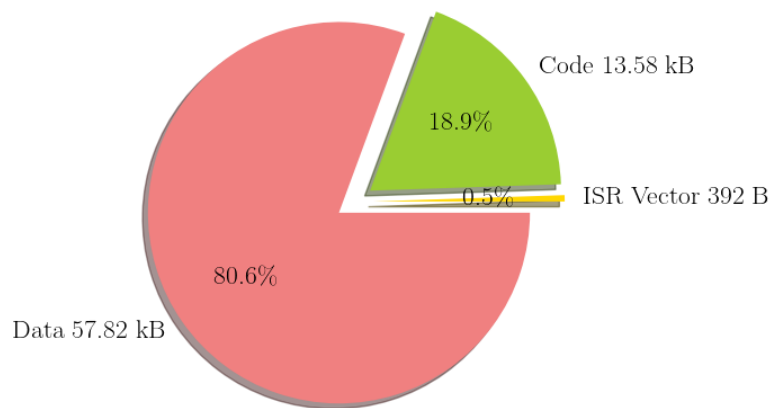


Table 44 – FLASH memory data distribution

Stage		FLASH memory (Bytes)
Wavelet Decomposition		32
Features Normalization		112
Prediction	Support Vectors	40320
	DualCoef.	17280
	Intercept	84
	N_SVperClass	14
	StartIdx	14
Total		57712
Other		1351
Total		59207

6.4 Conclusion

In this chapter, it is presented the embedded prototyping of the proposed Short Circuit Failure Detection algorithm, which corresponds to the last steps of this machine learning embedded application design. A computational resources optimization x Classification Performance Tradeoff Study is performed, the Model is Implemented in the Target Platform and is evaluated.

In the first one, the impact in classification accuracy and normal class sensitivity is evaluated when the total acquisition time and f_s are reduced considering the four feature sets described in the previous chapter. After evaluating the target platform specifications and a preliminary implementation of the DWT in the same device, the approach which performed better in the constrained context was chosen to be embedded, FS20 with TAT of 3 s and $f_s = 5$ kHz. Another brief analysis performed was the impact in classifier support vectors and binary accuracy if only normal and failure labels were considered. However, this last approach was presented only as backup option for a more constrained implementation case.

The algorithm developed and simulated in the workstation was successfully implemented in the target platform because there are no differences in the classification of 519 patterns in both platforms. The decrease in the 7 classes accuracy caused by the reduction in frequency and TAT is acceptable, because very good average 7 classes accuracy results still achieved, 97.26 %, and 99.92 % for 2 classes. The false alarm rate is only 0.2 %, with the number of points reduced from 100,000 to 15,000 and instead of 10 s of data acquisition, only 3 s. The response time, from multiresolution decomposition to classification takes only 104 ms, which is about 30 times smaller than the acquisition time. What makes the effort to reduce this time even further not very meaningful for this application. RAM memory usage can be optimized using Q15 data type, to allow the implementation in an even lower cost device.

Cunha et al. (2018) presents a study about the influence of the input data length, levels of decomposition and the filter length in the Wavelet decomposition execution time, RAM and FLASH footprint, using the library developed in this project. More information about the Naive Bayes and KNN implementation and metrics in the target platform are available in Cunha, Silva Jr and Medeiros (2018)'s paper.

7 Conclusion

This thesis describes a methodology to develop embedded applications which combine signal processing and machine learning techniques. The study case is the detection of incipient stator winding short-circuit failures in induction motors, using an axial flux signal.

The successful development of embedded solutions for two different classification problems using the described design methodology suggests it might be a good approach to similar problems. It reduces the chances of significant changes in the algorithm in the embedded implementation phase, improves the performance, and decreases computational resources usage.

This work represents a significant contribution to the development of an induction motor short-circuit failure detection equipment. Despite the variety of solutions already proposed by other authors, most of them are not evaluated in an embedded device and important aspects of signal acquisition and pre-processing are neglected. In this work, the complete model was successfully embedded to an STM32F407VGT6, from the wavelet decomposition to SVM classifier.

The good results obtained with the combination of MRA, statistical measurements and hybrid feature subset selection methods in two different embedded applications seem to indicate this approach can be applied to different problems involving machine learning and feature extraction from time domain signals. The inclusion of Multiresolution Analysis significantly minimizes false positive rate and also improves the classification performance in general. It also contributes to eliminate the FP errors and achieve very high binary accuracy.

The best classification results were obtained using the SVM classifier. OPF, MLP, ELM and KNN presented slightly worse results, while the other classifiers evaluated, RBF, Bayesian and Perceptron, presented poor results. The worst performance of the last classifier with linear or quadratic discriminant indicates this dataset is not linearly or quadratically separable.

Even for very incipient failure type, that only 1.41 % of the turns are shorted in a high impedance condition, 96.76 % of the failures are correctly classified. It shows how sensitive the model is for the detection of the short-circuit failures. Despite detecting very incipient faults, it doesn't imply in a classifier that tends to confuse normal samples with light faulty ones. The non-occurrence of FP supports this statement.

In the computational resources usage reduction phase, besides saving memory,

the optimization proposed contributed to decrease the processing and acquisition time, achieving much lower response time. It is important to this application and highlights the importance of this phase in the solution development process.

7.0.1 Future works

There are many possibilities to continue the study presented in this text. Considering the feature extraction phase, there are different mother wavelets and higher order wavelets to be evaluated trying to improve the feature set. Furthermore, an extension of MRA, the Wavelet Packet Decomposition, where the detail coefficients are also decomposed, splitting the signal in more frequency intervals, can be applied. It increases the resolution of the analysis and perhaps the failure detection can be improved further.

The analysis performed using the flux signal can be replicated to the three-phase current, which has already been acquired, for the detection of short-circuit and the results of both signals can be compared. Also, the detection of other defects in induction motors can be evaluated using the proposed approach or even other completely different applications.

The coil used to measure the axial leakage flux was placed inside the motor. However there are similar studies which the coil is placed outside and good results were also achieved. Thus, evaluate the results in this configuration is important to the proposition of non-invasive monitoring.

Other machine learning techniques can be applied or variations of the classifiers evaluated can be experimented seeking to improve the results or decrease computational resources usage. The analysis of the binary approach initiated during this work can be evaluated further as an option to more constrained and lower cost solutions.

The dataset used in this work was acquired from a modified machine where the failures are emulated. It would be interesting to evaluate the failure detection performance for real short-circuit data. As incipient conditions were well detected, it is expected the same happens to more severe conditions. Another interesting experiment is to evaluate the generalization capacity of this approach for higher-power motors.

Regard the embedded implementation, it was evaluated offline, using data previously acquired. The next step for the development of the short-circuit failure detection tool is to analyze the system online, acquiring and processing signals by the microcontroller, on a scenery closer to an industrial environment. Furthermore, some optimizations can be done to the implementation to decrease further the memory footprint and lower the cost.

As the total execution time is about 3.104 s only, the failure diagnostic could be released only after a few repetitive failure indications confirm the diagnostic. It can

contribute to increase the classifier robustness. Another approach to improve the robustness is to incorporate a rejection option, to safeguard against misclassifications, for patterns which the prediction is ambiguous, converting potential errors to rejection.

Bibliography

Analog Devices. *Basic Linear Design*. [S.l.]: Analog Devices, 2007. One citation in page 78.

Analog Devices. *Fixed-Point vs. Floating-Point Digital Signal Processing*. [S.l.], 2018. One citation in page 67.

ANDRA, K.; CHAKRABARTI, C.; ACHARYA, T. A VLSI architecture for lifting-based forward and inverse wavelet transform. *IEEE Transactions on Signal Processing*, Institute of Electrical and Electronics Engineers (IEEE), v. 50, n. 4, p. 966–977, apr 2002. One citation in page 67.

ARM. *Cortex-M4 Devices Generic User Guide*. [S.l.], 2011. ARM DUI 0553B. One citation in page 121.

BELLINI, A. et al. Advances in diagnostic techniques for induction machines. *IEEE Transactions on Industrial Electronics*, v. 55, n. 12, p. 4109–4126, Dec 2008. 3 citations in pages 32, 39, and 40.

BISHOP, C. *Pattern Recognition and Machine Learning*. [S.l.]: Springer-Verlag, Berlin, Heidelberg, 2006. 2 citations in pages 47 and 49.

BONNETT, A. H. Analysis of the impact pulse-width modulated inverter voltage waveform on ac induction motors. *IEEE Transactions on Industry Applications*, v. 32, n. 2, p. 386–392, March 1996. One citation in page 38.

BONNETT, A. H.; SOUKUP, G. C. Cause and analysis of stator and rotor failures in 3-phase squirrel cage induction motors. *IEEE Transactions on Industry Applications*, v. 28, n. 4, p. 921–937, July 1992. One citation in page 38.

BONNETT, A. H.; YUNG, C. Increased efficiency versus increased reliability. *IEEE Industry Applications Magazine*, v. 14, n. 1, p. 29–36, Jan 2008. One citation in page 32.

CABANAS, M. F. et al. Detection of stator winding insulation failures: On-line and off-line tests. In: *2013 IEEE Workshop on Electrical Machines Design, Control and Diagnosis (WEMDCD)*. IEEE, 2013. Disponível em: <<https://doi.org/10.1109/wemdc.2013.6525181>>. One citation in page 38.

CHANG YU-MING JHENG, C.-C. K. H.-C.; HSUEH, Y.-M. One citation in page 69.

CHERUKU, R.; EDLA, D. R.; KUPPILI, V. Diabetes classification using radial basis function network by combining cluster validity index and bat optimization with novel fitness function. *International Journal of Computational Intelligence Systems*, v. 10, 2017. One citation in page 56.

CORTES, C.; VAPNIK, V. Support-vector networks. *Machine Learning*, Springer Science and Business Media LLC, v. 20, n. 3, p. 273–297, sep. 1995. Disponível em: <<https://doi.org/10.1007/bf00994018>>. One citation in page 58.

COVER, T.; HART, P. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, Institute of Electrical and Electronics Engineers (IEEE), v. 13, n. 1, p. 21–27, jan. 1967. Disponível em: <<https://doi.org/10.1109/tit.1967.1053964>>. One citation in page 57.

CUNHA, R. G. et al. Toward an embedded application to identify degradation in energized polymeric insulators using discrete wavelet transform. In: *2018 25th International Conference on Systems, Signals and Image Processing (IWSSIP)*. [S.l.]: IEEE, 2018. 5 citations in pages 73, 121, 122, 124, and 129.

CUNHA, R. G. C.; Silva Jr, E. T. D.; MEDEIROS, C. M. D. S. An embedded application to identify degradation in energized polymeric insulators using machine learning and wavelet transform. In: *2018 VIII Brazilian Symposium on Computing Systems Engineering (SBESC)*. [S.l.]: IEEE, 2018. 2 citations in pages 129 and 146.

CUSIDÓ, J. et al. Fault detection in induction machines using power spectral density in wavelet decomposition. *IEEE Transactions on Industrial Electronics*, Institute of Electrical and Electronics Engineers (IEEE), v. 55, n. 2, p. 633–643, 2008. Disponível em: <<https://doi.org/10.1109/tie.2007.911960>>. One citation in page 40.

DREYFUS, G.; GUYON, I. Features extraction: Foundations and applications. In: _____. [S.l.]: Springer, 2006. cap. Assessment Methods, p. 65–88. One citation in page 46.

DUCH, W. In: GUYON, I. et al. (Ed.). *Features Extraction: Foundations and Applications*. 1. ed. Heidelberg: Springer, 2006. cap. Filter methods, p. 89–91. 2 citations in pages 46 and 102.

FALCAO, A. X.; STOLFI, J.; LOTUFO, R. A. The image foresting transform: Theory, algorithms, and applications. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, v. 26, n. 1, p. 19–29, jan. 2004. One citation in page 63.

FITZGERALD, A. E.; KINGSLEY, C.; UMANS, S. D. *Electric Machinery*. [S.l.]: The McGraw-Hill Companies, 2003. One citation in page 37.

FRIEDMAN, J. H. Regularized discriminant analysis. *Journal of the American statistical association*, 1989. One citation in page 50.

FROSINI, L. et al. A novel approach to detect short circuits in low voltage induction motor by stray flux measurement. In: *XXth International Conference on Electrical Machines*. Marseille, France: [s.n.], 2012. One citation in page 95.

FROSINI, L.; ZANAZZO, S.; ALBINI, A. A wavelet-based technique to detect stator faults in inverter-fed induction motors. *XXII International Conference on Electrical Machines (ICEM)*, 2016. 3 citations in pages 38, 39, and 40.

GARG, G. et al. Relative wavelet energy as a new feature extractor for sleep classification using eeg signals. *International Journal of Biomedical Signal Processing*, p. 75–80, 2011. One citation in page 45.

GIGAOM. Ai at the edge: A gigaom research byte. 1 2019. One citation in page 67.

GIRON-SIERRA, J. M. *Digital Signal Processing with Matlab Examples*. [S.l.]: Springer, 2017. Vol 1. One citation in page 40.

GUVEN, Y.; ATIS, S. Implementation of an embedded system for real-time detection of rotor bar failures in induction motors. *ISA Transactions*, Elsevier BV, v. 81, p. 210–221, oct 2018. One citation in page 69.

GUYON, I.; ELISSEEFF, A. In: GUYON, I.; NIKRAVESH, S. G. M.; ZADEH, L. A. (Ed.). *Features Extraction: Foundations and Applications*. 1. ed. Heidelberg: Springer, 2006. cap. An Introduction to Feature Extraction, p. 1–5. One citation in page 46.

HAAS, R.; DAVIES, J. *What's Powering Artificial Intelligence*. [S.l.], 2019. One citation in page 68.

HALL, M. A. Correlation-based feature selection for discrete and numeric class machine learning. In: *17th International Conference on Machine Learning*. San Francisco, CA, USA: [s.n.], 2000. p. 359–366. One citation in page 146.

HAYKIN, S. *Neural Networks and Learning Machines*. 3. ed. New Jersey, USA: Pearson Education, 2008. 10 citations in pages 48, 51, 53, 54, 56, 57, 58, 60, 61, and 86.

HUANG, G.-B.; ZHU, Q.-Y.; SIEW, C.-K. Extreme learning machine: Theory and applications. *Neurocomputing*, Elsevier BV, v. 70, n. 1-3, p. 489–501, dec. 2006. Disponível em: <<https://doi.org/10.1016/j.neucom.2005.12.126>>. One citation in page 54.

JAWADEKAR, A. et al. Artificial neural network-based induction motor fault classifier using continuous wavelet transform. *Systems Science & Control Engineering*, Informa UK Limited, v. 2, n. 1, p. 684–690, nov. 2014. Disponível em: <<https://doi.org/10.1080/21642583.2014.956266>>. One citation in page 40.

J.PENMAN. Detection and location of interturn short circuits in the stator windings of operating motors. *IEEE Transactions on Energy Conversion*, v. 9, n. 4, p. 652–658, dec. 1994. 3 citations in pages 39, 40, and 95.

JäGARE, U. *Embedded Machine Learning Design: Arm Special Edition*. [S.l.]: John Wiley & Sons, Inc., Hoboken, New Jersey, 2019. One citation in page 31.

KARIM, E.; MEMON, T. D.; HUSSAIN, I. FPGA based on-line fault diagnostic of induction motors using electrical signature analysis. *International Journal of Information Technology*, Springer Nature, v. 11, n. 1, p. 165–169, aug 2018. One citation in page 69.

Khan, T.; Alekhya, P.; Seshadrinath, J. Incipient inter-turn fault diagnosis in induction motors using cnn and lstm based methods. In: *2018 IEEE Industry Applications Society Annual Meeting (IAS)*. [S.l.: s.n.], 2018. p. 1–6. ISSN 2576-702X. One citation in page 41.

Kletter, D.; Messer, H. Suboptimal detection of non-gaussian signals by third-order spectral analysis. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, v. 38, n. 6, p. 901–909, June 1990. ISSN 0096-3518. One citation in page 42.

KOONS-STAPF, A. Condition based maintenance: Theory, methodology, application. In: *Reliability and Maintainability Symposium*. [S.l.: s.n.], 2015. One citation in page 33.

LAHOUD, N. et al. Electrical aging of the insulation of low-voltage machines: Model definition and test with the design of experiments. *IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS*, 2013. One citation in page 32.

- LANE, N. D. et al. Squeezing deep learning into mobile and embedded devices. *IEEE Pervasive Computing*, Institute of Electrical and Electronics Engineers (IEEE), v. 16, n. 3, p. 82–88, 2017. Disponível em: <<https://doi.org/10.1109/mperv.2017.2940968>>. One citation in page 41.
- LEE, D. T. L.; YAMAMOTO, A. *Wavelet Analysis: Theory and Applications*. [S.l.], 1994. Hewlett-Packard Journal. One citation in page 43.
- LEE, J.; SPANIAS, M. S. A.; TEPEDELENLIOGLU, C. Integrating machine learning in embedded sensor systems for internet-of-things applications. In: *IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*. [S.l.: s.n.], 2017. One citation in page 31.
- LEI, Y. *Intelligent Fault Diagnostic and remaining usefull life prediction of rotating machinery*. [S.l.]: Elsevier, 2017. 2 citations in pages 44 and 81.
- LIU, T.; MOORE, A. W.; GRAY, A. New algorithms for efficient high-dimensional nonparametric classification. In: SHAKHNAROVICH, G.; DARRELL, T.; INDYK, P. (Ed.). *Nearest-Neighbor Methods in Learning and Vision*. [S.l.]: The MIT Press, 2006. p. 75–101. One citation in page 57.
- MADRUGA, G. et al. A practical approach for detection of incipient failure of ceramic insulators. In: *IEEE International Conference on Solid Dielectrics*. Bologna, Italy: [s.n.], 2013. p. 182–185. One citation in page 145.
- MALLAT, S. *A Wavelet tour of signal processing: The sparse way*. [S.l.]: Elsevier, 2009. 3 citations in pages 43, 44, and 45.
- MALLAT, S. G. A theory for multiresolution signal decomposition: The wavelet representation. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, v. 11, n. 7, jul. 1989. One citation in page 44.
- MARTIN, H.; HONARVAR, F. Application of statistical moments to bearing failure detection. *Applied Acoustics*, Elsevier BV, v. 44, n. 1, p. 67–77, 1995. Disponível em: <[https://doi.org/10.1016/0003-682x\(94\)p4420-b](https://doi.org/10.1016/0003-682x(94)p4420-b)>. One citation in page 42.
- MARWEDEL, P. *Embedded Systems Design*. [S.l.]: Springer, 2006. 2 citations in pages 65 and 66.
- MCCORMICK, C. Radial basis function network (rbfn) tutorial. <https://mccormickml.com/2013/08/15/radial-basis-function-network-rbfn-tutorial/>. 2013. One citation in page 55.
- MCCULLOCH, W. S.; PITTS, W. H. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, v. 5, p. 115–133, dec 1943. One citation in page 48.
- MCLAUGHLIN, S.; STOGIOGLOU, A.; FACKRELL, J. Introducing higher order statistics (hos) for the detection of nonlinearities. Department of Electrical Engineering, University of Edinburgh. 1995. One citation in page 42.
- MENON, R. et al. (Ed.). *Correlation of acoustic emission method and electrical method for detection of partial discharges in transformers*. Eindhoven, Netherlands: IEEE 7th International Conference on Solid Dielectrics, 2001. One citation in page 145.

MERRY, R. J. E. *Wavelet theory and applications: a literature study*. Dissertação (Mestrado) — Eindhoven University of Technology, 2005. One citation in page 96.

MITCHELL, T. M. *Machine Learning*. [S.l.]: McGraw-Hill, 1997. One citation in page 47.

MULGREW, B.; GRANT, P.; THOMPSON, J. *Digital Signal Processing Concepts and Applications*. 2nd edition.ed.[S.l.] : PalgraveMacmillan, 2003. One citation in page 40.

NASCIMENTO, N. M. M. *Detecção e classificação de falhas de curto-circuito em geradores de indução aplicados em geração eólica utilizando aprendizado de máquina*. Dissertação (Mestrado) — INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO CEARÁ, 2018. 2 citations in pages 75 and 77.

OLIVEIRA, A. G. *Detecção e classificação de falhas de curto-circuito em geradores de indução aplicados em geração eólica utilizando aprendizado de máquina*. Dissertação (Mestrado) — Federal UNiversity of Ceará, 2014. One citation in page 94.

OLIVEIRA, G.; SILVA, R. S. T.; MEDEIROS, C. M. S. Neural network used to stator winding inter-turns short-circuit fault detection in an induction motor driven by frequency converter. *Proceedings - 1st BRICS Countries Congress on Computational Intelligence*, v. 1, p. 459–464, 2013. 4 citations in pages 33, 40, 41, and 74.

ORDAZ-MORENO, A. et al. Automatic online diagnosis algorithm for broken-bar detection on induction motors based on discrete wavelet transform for FPGA implementation. *IEEE Transactions on Industrial Electronics*, Institute of Electrical and Electronics Engineers (IEEE), v. 55, n. 5, p. 2193–2202, may 2008. One citation in page 67.

PANIGRAHY, P. S. et al. FPGA friendly fault detection technique for drive fed induction motor. In: *2016 2nd International Conference on Control, Instrumentation, Energy & Communication (CIEC)*. [S.l.]: IEEE, 2016. One citation in page 69.

PAPA, J. P. et al. Design of robust pattern classifiers based on optimum-path forests. *8th International Symposium on Mathematical Morphology*, p. 10–13, nov 2007. One citation in page 63.

PENROSE, R. A generalized inverse for matrices. *Mathematical Proceedings of the Cambridge Philosophical Society*, Cambridge University Press, v. 51, n. 3, p. 406–413, 1955. One citation in page 57.

RAITOHARJUAND, J.; KIRANYAZ, S.; GABBOUJ, M. Training radial basis function neural networks for classification via class-specific clustering. *IEEE Transactions on Neural Networks and Learning Systems*, v. 27, p. 1–14, 12 2015. One citation in page 56.

Reality AI. *The 2018 Ultimate Guide to Machine Learning for Embedded Systems*. 2018. One citation in page 68.

Rebouças Filho, P. P. et al. A reliable approach for detection of incipient faults of short-circuits in induction generator using machine learning. *Computers and Electrical Engineering*, v. 71, p. 440–451, oct 2018. 2 citations in pages 40 and 74.

RIERA-GUASP, M.; ANTONINO-DAVIU, J. A.; CAPOLINO, G.-A. Advances in electrical machine, power electronic and drive condition monitoring and fault detection: State of the art. *IEEE Transactions on Industrial Electronics*, v. 62, n. 3, p. 1746–1759, March 2015. 4 citations in pages 32, 37, 38, and 39.

ROCKWELL AUTOMATION. *Application basics of operation of three-phase induction motors*. [S.l.], 1996. 2 citations in pages 37 and 38.

RODRIGUEZ-DONATE, C. et al. FPGA based embedded system for induction motor failure monitoring at the start-up transient vibrations with wavelets. In: *2008 International Symposium on Industrial Embedded Systems*. [S.l.]: IEEE, 2008. One citation in page 69.

ROSENBLATT, F. The perceptron – a perceiving and recognizing automaton. *Technical Report 85-460-1, Cornell Aeronautical Laboratory*, 1957. One citation in page 48.

ROSENBLATT, F. The perceptro: A probabilistic model for information storage and organization in the brain. *Psychological Review*, v. 65, n. 6, 1958. 2 citations in pages 48 and 51.

SAMANTA, B.; AL-BALUSHI, K. ARTIFICIAL NEURAL NETWORK BASED FAULT DIAGNOSTICS OF ROLLING ELEMENT BEARINGS USING TIME-DOMAIN FEATURES. *Mechanical Systems and Signal Processing*, Elsevier BV, v. 17, n. 2, p. 317–328, mar. 2003. Disponível em: <<https://doi.org/10.1006/mssp.2001.1462>>. One citation in page 42.

SHAO, S.-Y. et al. A deep learning approach for fault diagnosis of induction motors in manufacturing. *Chinese Journal of Mechanical Engineering*, Springer Nature, v. 30, n. 6, p. 1347–1356, oct. 2017. Disponível em: <<https://doi.org/10.1007/s10033-017-0189-y>>. One citation in page 41.

SIFUZZMAN, M.; ISLAM, M. R.; ALI, M. Z. Application of wavelet transform and its advantages compared to fourier transform. *Journal of Physiscal Sciences*, 2009. One citation in page 95.

SOUSA, I. R.; MEDEIROS, C. M. S. Aplicação de redes neurais artificiais para detecção de falhas em barras rotóticas de motores de indução. *Símpósio Brasileiro de Automação Inteligente*, 2017. One citation in page 76.

SOUSA, P. H. F. et al. Detection and classification of faults in induction generator applied into wind turbines through a machine learning approach. *International Joint Conference on Neural Networks (IJCNN)*, 2018. 2 citations in pages 76 and 94.

SOUZA, R.; LOTUFO, R.; RITTNER, L. Comparison between optimum-path forest and k-nearest neighbors classifiers. *SIBGRAPI Conference on Graphics, Patterns and Images*, 2012. 2 citations in pages 63 and 64.

STEFENON, S. F.; MEYER, L. H.; MOLINA, F. H. Real time automates diagnosis of insulating system employing ultrasound inspection. In: *23rd International Conference on Electricity Distribution*. Lyon: [s.n.], 2015. One citation in page 145.

STMICROELECTRONICS. *AN4044 Application note: Floating point unit demonstration on STM32 microcontrollers*. [S.l.], 2016. 2 citations in pages 67 and 121.

- STMICROELECTRONICS. *AN4841 Application note: Digital signal processing for STM32 microcontrollers using CMSIS*. [S.l.], 2018. 2 citations in pages 66 and 124.
- SUGIYAMA, M. *Introduction to Statistical Machine Learning*. [S.l.]: Elsevier, 2016. One citation in page 42.
- SURYA, G. N. et al. A simplified frequency domain detection of stator turn fault in squirrel cage induction motors using observer coil technique. *IEEE Transactions on Industrial Electronics*, v. 64, n. 2, p. 1495–1506, 2017. One citation in page 39.
- TAN, L. *Digital Signal Processing Fundamentals and Applications*. Decatur, Georgia, USA: Elsevier, 2008. One citation in page 77.
- TAVNER, P.; PENMAN, J. *Condition Monitoring of Electrical Machines*. [S.l.]: Research Studies Press, 1987. One citation in page 39.
- TEKTRONIX. *Mixed Signal Oscilloscopes MSO4000B, DPO4000B Series Datasheet*. [S.l.], 2016. One citation in page 78.
- THOMSON, W. T. Diagnosing faults in induction motors - engineering ideas. *Electrical Review*, 1984. One citation in page 39.
- THOMSON, W. T.; GILMORE, R. J. Motor current signature analysis to detect faults in inudction motor drives - fundamentals, data, interpretation, and industrial case histories. *Motor Current Signature Snalysis to Detect Faults in Inudction Motor Drives - Fundamentals, Data, Interpretation, and Industrial Case Histories.*, 2003. One citation in page 39.
- UKIL, A.; BARLOCHER, A. Implementation of discrete wavelet transform for embedded applications using TMS320vc5510. In: *2007 International Symposium on Industrial Embedded Systems*. [S.l.]: IEEE, 2007. 2 citations in pages 67 and 124.
- VENKATARAMANI, B.; BHASKAR, M. *Digital Signal Processors: Architecture, Programming and Applications*. Second edition. [S.l.]: Tata McGraw-Hill, 2011. One citation in page 66.
- VIEIRA, R. G. et al. Embedding a neural classifier to detect faults in a three-phase induction motor. *2016 VI Brazilian Symposium on Computing Systems Engineering*, p. 136–143, Nov 2016. 3 citations in pages 33, 41, and 69.
- VIEIRA, R. G.; MEDEIROS, C. M. S.; SILVA, E. T. Classification and sensitivity analysis to detect fault in induction motors using an MLP network. In: *2016 International Joint Conference on Neural Networks (IJCNN)*. [S.l.]: IEEE, 2016. 2 citations in pages 90 and 93.
- WANG, A. et al. Accelerating wrapper-based feature selection with k-nearest-neighbor. *Knowledge-Based Systems*, v. 83, july 2015. One citation in page 96.
- WANG, Z.; XUE, X. Multi-class support vector machine. In: *Support Vector Machines Applications*. Springer International Publishing, 2014. p. 23–48. Disponível em: <https://doi.org/10.1007/978-3-319-02300-7_2>. One citation in page 62.

WEG. *CFW-08 User's Guide*. [S.l.], 2019. 0899.5242 / 09. Disponível em: <<https://static.weg.net/medias/downloadcenter/hf7/hc8/WEG-cfw08-users-manual-0899.5242-manual-en.pdf>>. One citation in page 73.

WEISER, M. Hot topics - ubiquitous computing. *Computer*, p. 71–72, oct. 1993. One citation in page 31.

WOLF, M. *High-Performance Embedded Computing : Applications in Cyber-Physical Systems and Mobile Computing*. [S.l.]: Morgan Kaufmann Publishers, Elsevier, 2014. One citation in page 66.

WOLF, W. H. *Computers as Components: Principles of Embedded Computing System Design*. [S.l.]: Morgan Kaufmann Publishers, Elsevier, 2008. 2 citations in pages 31 and 65.

WOLKIEWICZ, M.; SKOWRON, M.; KOWALSKI, C. T. Electrical fault diagnostic system based on the kohonen neural network classifier. In: *2018 International Symposium on Electrical Machines (SME)*. IEEE, 2018. Disponível em: <<https://doi.org/10.1109/isem.2018.8442673>>. One citation in page 40.

YAZIDI, A. et al. Flux signature analysis: An alternative method for the fault diagnosis of induction machines. *IEEE Russia Power Tech*, p. 1–6, June 2005. One citation in page 39.

YAZIDI, A. et al. Detection of stator short-circuit in induction machines using an external leakage flux sensor. *IEEE International Conference on Industrial Technology(ICIT)*, v. 1, p. 166–169 Vol. 1, Dec 2004. One citation in page 94.

YHDC. *Split core current transformer*. [S.l.], 2010. Disponível em: <en.yhdc.com/comp/file/download.do?id=947>. Acesso em: 17 may. 2019. One citation in page 76.

Zheng Liu et al. Non-invasive winding fault detection for induction machines based on stray flux magnetic sensors. *IEEE Power and Energy Society General Meeting (PESGM)*, p. 1–6, July 2016. One citation in page 39.

Appendix

APPENDIX A – Energized Polymeric Insulator Degradation Detector: Another case study

The same methodology described in section 3 was applied to the problem of identifying degradation in energized polymeric insulators. The same signal processing technique described in this dissertation was also applied: The Wavelet Multiresolution Analysis combined to statistical features.

Failures in insulators may lead to damaging consequences to the electric power system. Those vary from interferences in radio communications to temporary interruption of supply, that might provoke losses to the consumer units and the energy companies. Among several insulator inspection techniques, those based on acoustic emissions are getting momentum, due to their reduced price and immunity to electromagnetic noise (MENON et al., 2001).

These defects have many causes, like surface contamination by pollutants, cracks and structural failures not visually noticeable. Electrical discharges in defective insulators produce sound waves that carry information about the cause of the defect (STEFENON; MEYER; MOLINA, 2015). This way, those signals can be acquired and processed to predict the insulator status. Some energy distribution companies already use ultrasound to perform regular inspections in the network, but the decision of replacing or not the inspected device depends on the operator perception and its experience with the equipment (MADRUGA et al., 2013).

Thus, it would be helpful to have an automatic tool to indicate the degradation in those insulators, providing resources to the maintenance team to plan cleaning and replacement routines. An equipment to automatically indicate the degradation can be attached to an electric tower. In this case, it should be logically connected to a large diagnosis system, sending messages periodically (typically 1 minute). Another approach could be a portable device, used for routine inspection. In this case, an acceptable latency for diagnostic could be 0.5 seconds. Both cases should be energy-aware since they are powered by batteries.

In this application ultrasound signals are used to identify different types of pollutants in energized polymeric insulators. The dataset used in this work contains 1464 patterns, divided into 5 classes: the normal condition (without corona effect), and 4 degraded ones; oxidized, with bird feces, salty pollution, and flash-over (moderated discharges). Each

pattern is composed of 5000 points, resulting in 22 ms in total. As the network frequency is 60 Hz, each sample is a little larger than a full cycle of 16.67 ms. This is necessary because the ultrasound signal emitted by the insulator happens at least once in a cycle. To capture the signals a 12 bit ADC was used with 225 kHz as the sampling frequency. The ultrasonic signals were filtered to remove noise outside the frequency band of interest, 40 kHz, and amplified to the voltage level compatible with the analog-to-digital converter range (0 V to 3.3 V).

Based on the characteristics of the insulator ultrasound signal, using the Wavelet transform should be a better approach than apply the Fourier transform, because the signal is non-periodic and presents peaks.

Five levels of multiresolution decomposition using DWT with the Daubechies-4 mother wavelet was performed. Statistical features were calculated from the approximation and detail coefficients. To reduce the number of feature, keeping only the most representative ones, Correlation Features Selection (CFS) was used. It evaluates the feature merit based on the relevance to represent a class and also on the correlation level between the features (HALL, 2000). None of the features of the levels 1 and 2 and from the original signal were selected using this method. From detail coefficients level 3 (D3), percent energy and standard deviation were chosen. For level 4, percent energy, standard deviation and mean absolute deviation. For detail coefficients level 5, percent energy and mean absolute deviation. For approximation coefficients level 5, percent energy standard deviation and mean absolute deviation were selected. The feature set size was reduced to 10 features.

The classifiers analyzed were OPF, SVM, KNN and Naive Bayes (CUNHA; Silva Jr; MEDEIROS, 2018). As the results were all very close, the two simplest classifiers, Naive Bayes and KNN were chosen to be evaluated in the target platform. The Naive Bayes presented $91.72\% \pm 0.64$ global mean accuracy and Knn, $93.31\% \pm 0.66$.

Table 45 summarizes the execution time and memory footprint in the target platform using Naive Bayes and Knn. The last one has better accuracy, but it has higher execution time and memory footprint. This application is not time critical and the selected platform has enough memory to run the pre-processing and the classification even for Knn, with 80 % of the dataset. This way, Knn should be the best option. Similar to the main application described, the biggest part of the execution time corresponds to the features extraction routine, that takes 35,095 μs .

Table 45 – Summary of the results. Training set = 80 % of the dataset, Dataset 3, K=5, FPU enabled

	Execution Time (μs)	RAM (bytes)	FLASH (bytes)
Features Extraction +Naive Bayes	36,529	43,012	18,812
Features Extraction +Knn	43,420	47,544	64,704