UNIVERSIDADE FEDERAL DA BAHIA – UFBA

DEPARTAMENTO DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO - DEEC

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA - PPGEE

Lucas Cruz da Silva

DECISION SUPPORT SYSTEM FOR ULTRASOUND NON-DESTRUCTIVE EVALUATION BASED ON EXTREME LEARNING MACHINES EMBEDDED IN MICROCONTROLLERS

DOCTORAL THESIS

Salvador
July 2020

PPGEE/UFBA

DECISION SUPPORT SYSTEM FOR ULTRASOUND NON-DESTRUCTIVE EVALUATION BASED ON EXTREME LEARNING MACHINES EMBEDDED IN MICROCONTROLLERS

DOCTORAL THESIS

Lucas Cruz da Silva

Thesis presented to the Electrical Engineering Graduate Program (PPGEE/UFBA) in partial fulfillment of the requirements for the degree of Doctor in Electrical Engineering.

Advisor: Eduardo F. de Simas Filho, D.Sc.

Salvador

July 2020

S586 Silva, Lucas Cruz da.
Decision support system for ultrasound non-destructive evaluation based on extreme learning machines embedded in microcontrollers / Lucas Cruz da Silva. – Salvador, 2020.
130 f. : il. color.

Orientador: Prof. Dr. Eduardo F. de Simas Filho.

Tese (doutorado) – Universidade Federal da Bahia. Escola Politécnica, 2020.

1. Tomada de decisão. 2. Microcontroladores. 3. Avaliação não destrutiva. 4. Ultrassom. 5. Redes neurais artificiais. I. Simas Filho, Eduardo F. de. II. Universidade Federal da Bahia. III. Título.

CDD.: 658.402

Lucas Cruz da Silva

" Decision Support System for Ultrasound Non-Destructive Evaluation based on Extreme Learning Machines Embedded in Microcontrollers"

Tese apresentada à Universidade Federal da Bahia, como parte das exigências do Programa de Pós-Graduação em Engenharia Elétrica, para a obtenção do título de Doutor.

APROVADA em: 08 de Julho de 2020.

BANCA EXAMINADORA

Prof. Dr. Eduardo Furtado de Simas Filho
Orientador/UFBA

Prof. Dr. Romis Ribeiro de Faissol Attux
UNICAMP

Prof. Dr. Luciano Manhães de Andrade Filho
UFJF

Prof. Dr. Paulo Cesar Machado de Abreu Farias
UFBA

Prof. Dr. Antonio Carlos Lopes Fernandes Jr
UFBA

Acknowledgements

I would like to thank FAPESB for the financial support and GPEND-IFBA for the technical support and infrastructure availability.

I wish to recognize the invaluable assistance of my advisor Prof. Eduardo Simas, whose knowledge, wisdom and thorough support is a reference and inspiration to me and made this work a pleasant journey. I also appreciate the experience shared with my colleagues during this work.

Lastly, I would like to express my gratitude to my family Paulo, Miriam and Júlia for the enthusiastic support. I deeply thank my wife Rebeca for the limitless patience, love and support.

[...] That's all it is. You just begin. You do the math. You solve one problem... and you solve the next one... and then the next. And If you solve enough problems, you get to come home.

— Mark Watney, character from The Martian motion picture (2015)

Abstract of Thesis presented to PPGEE/UFBA as partial fulfillment of the requirements for achieving the Doctor in Electrical Engineering degree.

DECISION SUPPORT SYSTEM FOR ULTRASOUND NON-DESTRUCTIVE EVALUATION BASED ON EXTREME LEARNING MACHINES EMBEDDED IN MICROCONTROLLERS

Lucas Cruz da Silva

July/2020

Advisor: Eduardo F. de Simas Filho, D.Sc.

Graduate Program: Electrical Engineering Graduate Program

This work aims to design and implement an intelligent ultrasound non-destructive evaluation decision support system, that may be portable to a microcontrolled platform. Extreme learning machine, a fast training class of artificial neural network, was used as pattern classifier. The purpose is to obtain a dedicated system capable of recording the input data, training and operating the classifier system, and finally producing an integrity status indication for supporting the operator decision. A case study on integrity evaluation of weld beads in steel plates was considered. The basics of ultrasonic non-destructive testing and artificial neural networks are introduced. Further aspects are considered in order to accomplish the referred objective, such as: a comparison between a fast-training neural network and traditional ones; a new segmentation approach for analysis of time-of-flight diffraction signals; a comparison between methods to reduce the feature set dimensionality and the proposed approach for system realization in microcontrollers, which comprises the evaluation of random weights generation and quantization in extreme learning machines. The results indicate that the system realizes the training step of a classifier in the microcontroller to carry out the classification of weld defects in similar accuracy performance as reported works in the literature.

Contents

List of Figures

List of Tables

Symbols and Abbreviations

Symbols

b	Bias of a neuron
B	Bit count
C	Regularization constant
D	Diameter of ultrasonic probe
E	Error energy of a neuron
f	Frequency of a mechanical wave
$\mathbf{H}$	Hidden layer output
$\mathbf{H}^{\dagger}$	Moore-Penrose pseudoinverse of $\mathbf{H}$
H_0	Negative hypothesis
H_1	Positive hypothesis
k	Shape parameter in Weibull probability density function
l	Length of 32-bit integer array
L	Neuron count in hidden layer
n	Node count in input layer
N	Samples count
NF	Near field distance
m	Neuron count in output layer
p	Acoustic pressure
R	Ratio of reflected energy in an interface
$\mathbf{t}$	Target vector
T	Test samples count
TA	Test Accuracy
U	Particle velocity

V	Speed of mechanical waves in a specimen
W	Random weight count
$\mathbf{w}$	Vector of weights of a neuron
$\mathbf{x}$	Input vector
Z	Acoustic Impedance
$\boldsymbol{\beta}$	Matrix containing weights of output layer
γ	Scale parameter in Weibull probability density function
δ	Local gradient of a neuron
η	Learning rate
λ	Wavelength of a mechanical wave
Λ	Constant in Exponential probability density function
μ	Mean of Normal (Gaussian) probability density function
ρ	Density of a specimen
σ	Standard deviation of Normal (Gaussian) probability density function
θ	Incidence angle of ToFD ultrasonic probe

Abbreviations

ABENDI	Associação Brasileira de Ensaios Não-Destrutivos e Inspeção
ABNT	Associação Brasileira de Normas Técnicas
ADC	Analog to Digital Converter
ALU	Arithmetic and Logic Unit
ANN	Artificial Neural Networks
API	Application Programming Interface
ASNT	American Society for Nondestructive Testing
BLAS	Basic Linear Algebra Subprograms
CDF	Cumulative Distribution Function
CPU	Central Processing Unit
DCT	Discrete Cosine Transform
DFT	Discrete Fourier Transform

DSP	Digital Signal Processor
ELM	Extreme Learning Machine
ELM-AE	Extreme Learning Machine Autoencoder
FPGA	Field programmable gate array
GMAW	Gas-Metal Arc Welding
GSL	GNU Scientific Library
GTAW	Gas-Tungsten Arc Welding
I/O	Input/Output
LF	Lack of Fusion
LM	Levenberg-Marquardt
LMS	Least Mean Squares
LP	Lack of Penetration
MAC	Multiply and Accumulate
MCU	Microcontroller
MFLOPS	Millions of Floating Point Operations per Second
MIPS	Millions of Instructions per Second
MSE	Mean Square Error
ND	No Defect
NDE	Non-Destructive Evaluation
NDT	Non-Destructive Testing
PC	Personal Computer
PCA	Principal Component Analysis
PDF	Probability Density Function
PLD	Programmable Logic Device
PO	Porosity
ReLU	Rectified Linear Unit
RPROP	Resilient Backpropagation
SI	Slag Inclusion
SMAW	Shielded-Metal Arc Welding
ToFD	Time-of-Flight Diffraction

Chapter 1

Introduction

1.1 Motivation

Non-destructive evaluation (NDE, also known as non-destructive testing, NDT) is performed by examination of an object without altering its form in order to search for discontinuities, flaws, or conditions that could compromise the overall serviceability of equipment and structures [1]. It is a key method in industry that provides a way to evaluate the conditions of raw materials prior to productions, to investigate the integrity of produced specimens without having to compromise their structure, and to assess the actual conditions of products and structures already under work (such as machines, industrial ovens, pipelines, etc). The applicability of such paradigm is within the interest of several technological areas, for instance oil and gas [2, 3], military/defense [4], construction [5], microelectronics [6] and so on.

Many techniques are employed in NDE, and some examples of the major methods used in industry are listed below [7]:

- **Visual**. The most known to human daily life, it relies on visual characteristics of an object for further evaluation by humans. Moreover, it can be also based on emitted light from objects captured by a light sensor, with further signal processing.

- **Eddy current**. This method uses electromagnetic fields induced in conductive specimens, producing electric currents (eddy currents) that appear as a result of the electromagnetic interaction. The change of these currents' char-

acteristics along the specimen can reveal defects.

- **Radiographic**. It uses incident radiation at the specimen, which is placed between the radiation source and a radiographic film. The film exposure to the radiation can be affected by discontinuities inside the specimen, which can be analyzed according to the record in the film.

- **Ultrasonic**. This method employs mechanical waves travelling along the material to be evaluated. Flaws and discontinuities may cause detectable reflections and diffractions.

Ultrasonic evaluation is in many cases preferable over the others because it provides information about potential flaws inside the material, it does not have inherent hazardous nature, like the radiographic method, and it is better than electrical tests for evaluating flaws deep farther from the specimen's surface [8]. Ultrasonic testing also has the advantage of being suitable for almost all solid thick media, summed up with large commercial availability for instruments [8]. However, this approach has drawbacks, for example when it is required to detect defects near beneath surfaces because this is an insensitive region due to restrict ultrasound beam width [9].

The evaluation is usually carried out by qualified inspectors [1]. Varying according to the employed technique, certain measures of observable characteristics are used to determine whether a specimen can be put into service. A decision is made by the inspector upon gathering the available information, backed up by training and qualification. However, several phenomena may mislead the inspector to a wrong interpretation regarding the condition of the evaluated specimen. Considering this, inspectors may benefit from a portable electronic system capable of providing valuable decision support information.

The state of the art in current methods within the field of artificial intelligence shows that the task of pattern recognition has spread among several applications in science and technology [10]. A method that has been playing a major role in this context is the one known as Artificial Neural Networks [11]. In its traditional approach, such technique is capable of creating a classifier system by training itself to recognize patterns within a given sample set by associating input-output pairs (supervised training) [12]. Once the training procedure is done, the system can

classify new inputs based on what it learned so far. This kind of supervised training method can be conveniently added to the task of evaluation done by inspectors, since it relies on previous knowledge about which class a certain sample belongs to.

Therefore, the task of evaluating a specimen's integrity can be more easily accomplished with the aid of decision support systems, considering that the data acquired from specimens in evaluation usually present specific patterns. In order to build a decision support system, it is required to guarantee that such system identify correctly the actual state of the specimen under evaluation and lead the inspector to the right diagnosis.

Often, the signals acquired from a specimen are submitted to some sort of preprocessing, which is meant to extract relevant features from the signal that will serve as input to the classification system. Many methods for extracting relevant features from sample sets are established in the areas of digital signal processing and statistical signal processing, among them one can mention deterministic (Fourier, wavelet and cosine discrete transforms) [13] and statistical (principal component analysis, and independent component analysis) methods[12, 14], so that systems built with trained classifiers can be greatly improved by these methods once the relevant features for classification are extracted from raw NDE data.

Many works follow the methodology presented in [15, 16, 17], which employs artificial intelligence to build decision support systems on desktop computer environments. This way, one must collect data at the specimen's location (usually in industry) and transfer it to a computer environment to train the system. The dependency on computer environment to train the system means that every NDE application may require:

1. A data acquisition interface, proper for acquiring data at the specimen and transferring this data to a computer with compatible data format and interfaces.

2. A computer environment with installed software capable of training classifiers.

Those aspects contribute to reduce the practical application of decision support systems for in loco specimen evaluation. In this sense, a portable device capable of performing signal processing and pattern classification is of great interest for NDE applications.

In recent years, microcontrollers have been increasing their share in the market of microprocessed solutions. With more models appearing each year comprising higher processing power, memory and a large number of interfaces, certain tasks of lower complexity (compared to the ones in a personal computer context) can be carried out by these devices instead of being executed by personal computers, which were for a long time the paramount tool for numerical and logical tasks. Platforms run by microcontrollers, often referred to as embedded devices, can be responsible for automation tasks, interface with other devices, local data network management, etc. Therefore, microcontrollers can be applied in the decision support system for NDE. In fact, many works shown in section 3.6.2 ported trained artificial neural networks to microcontrollers to act as classifiers in embedded systems. Nevertheless, microcontrollers face many technical constraints in the training procedure of traditional artificial neural networks, which is why a computer is still needed in this step. In this regard, the present work proposes to build a decision support system that can be miniaturized into an instrument or robot, having the capability to collect data at the specimen's location and to execute the system training procedure.

1.2 Scope of work

This work has the purpose of designing and implementing a decision support system for defect evaluation in weld beads joining steel plates, using ultrasonic nondestructive testing. The proposed system should be portable to an embedded platform, such as an instrument or a robot. The system design allows it to be employed in other ultrasonic NDE contexts as well. The proposed approach consists in providing known information of the specimen‘s condition to the embedded decision support system that allows it to train itself in order to estimate the quality of the inner physical structure of specimens. The mentioned training task shall take place in the embedded system, without using personal computers containing specialized frameworks to execute the training step.

The scope of this work is restricted to the training and identification of certain types of defects found commonly in welding processes. For the purpose of training and identification tasks, each defect (and also the absence of defects) will be cate-

gorized into classes, and the entire approach involving artificial intelligence will care only about the type of the defects. No measure nor defect sizing inside the weld bead will be considered in this work. Moreover, this work will not deal with data acquisition and manipulation at the microcontroller level, being restricted to the conception of the classifier in the device.

The specific objectives and contributions of this work lie on the three fields of knowledge listed in the following.

- **Feature selection**. The Time-of-Flight Diffraction (ToFD) ultrasonic technique was the source of signals forming the raw data from NDE to be provided as sample set to the embedded classifier training procedure. The signals were analyzed in both time (following a proposed approach for signal segmentation) and frequency domains in order to provide relevant features with low dimensionality. Feature extraction techniques such as principal component analysis and autoencoders were also evaluated for dimensionality reduction. In this context, the main contributions of this thesis are a novel proposal for time-domain segmentation of ToFD signal analysis and the estimation of a compact and relevant features set.

- **Artificial Intelligence**. The embedded platform will perform classification tasks based on Artificial Neural Networks. A special theory for training networks called Extreme Learning Machine was employed, which presents an advantageous training procedure for use in embedded platforms. This thesis contributes with an optimized approach regarding memory usage for porting extreme learning machines to embedded platforms.

- **Embedded platform**. A setup to an environment in a microcontrolled platform for pattern learning and recognition was developed in this work. The system was built on a STM32F407 microcontroller, an ARM cortex-M4 driven unit. The algorithms for training and classifying samples will be coded in `C++` language with the support of GNU Scientific Library ported for algebraic operations ported to ARM 32 bit processor, using cross compilation from Linux 64-bit environment. Tests will be executed regarding the system's accuracy performance and training time.

1.3 Document Organization

This document is organized as follows. Chapter 2 contains an overview of welding process and Non-destructive Evaluation focusing in ultrasonic techniques, specially the one used in this work, Time-of-Flight Diffraction. In Chapter 3 the definition of Artificial Neural Network is presented, with particular interest in Extreme Learning Machines. Also, a short discussion on microcontrollers and their implementations in the field of Artificial Neural Networks is presented. Chapter 4 contains the description of the data set used in this work and the proposed methodology to evaluate the experiments described in the following two chapters, which contain the outcomes of this work. Chapter 5 presents a comparison between Extreme Learning Machines and backpropagation-based methods to train Artificial Neural Networks. Moreover, a new proposal for frequency content analysis of ultrasonic data is introduced. Chapter 6 addresses aspects of implementation of Extreme Learning Machines in the microcontroller, such as input data dimensionality, random weight function generator evaluation and weight discretization. Finally, Chapter 7 presents the conclusions, lists contributions of this work and suggestions for research inspired in the presented results. In Appendix A the ELM implementation developed for this work is detailed, and Appendix B summarizes the elaborated papers during this work's period.

Chapter 2

Ultrasonic Non-Destructive Evaluation

2.1 Non-destructive evaluation in industry

NDE methods are widely employed in industry. The proper NDE method choice is usually driven by financial and safety constraints, and also considering the target application, which should be, for example, process quality control, predictive maintenance or material characterization [7]. Process and quality control are employed in metallurgy as a mean of process feedback, when the product must return to the production line in order to be compliant to a certain standard. In food industry, NDE methods are used as accept/reject criteria upon finishing the product. The difference between these two approaches is justified by economic reasoning: in the former, NDE being placed in the middle of the process produces less cost than having to return a finished product to the production line, whereas for the latter the cost of production evaluation may be dampened by scalability when put in the process end [7].

In terms of maintenance, transport industry (e.g. aerospace) and segments that deal with infrastructure (power and construction industries and extraction of natural resources such as oil & gas industry) employ NDE in their assets [7]. Since these segments work with core activities and assets whose interruption or ceasing activities might result in great economic or human life losses in many other segments, the NDE methods employed seek to inspect the conditions of the involved parts in hope to

early identify compromised structures or otherwise assure their serviceability and estimate the useful lifetime of its components [7].

The aforementioned industry segments employ NDE in valuable assets, and therefore require trained personnel to run the evaluation. As the many NDE methods evolved through the 20th century, organizations started trying to standardize these methods in order to qualify inspectors to better achieve industry needs. Several organizations have standards and provide certification for non-destructive evaluation in their respective countries, as is the case for the American Society for Nondestructive Testing (ASNT) [18] in USA. In Brazil, the Brazilian Association of Non-Destructive Testing and Inspection (ABENDI) [19] provides certification and coordinates technical events, while the Brazilian Association of Technical Standards (ABNT) [20] provides standards based on the international standard ISO 9712, which is reviewed every five years and establishes directives for qualification of personnel in industry to perform many NDE methods.

Nondestructive evaluation currently comprises a collection of different methods. A comprehensive summary table of the major NDE methods can be found in [7], which is partially reproduced in Table 2.1, showing more methods in addition to the ones mentioned in Section 1.1, along with some of their applications, advantages and limitations.

Lastly, although the usage of non-destructive evaluation looks advantageous in several industry segments, it is important to mention that NDE should not be considered as a next-level approach to destructive testing. Both paradigms are important for science of materials and technology and coexist in industry [7]. Destructive tests alter the specimen under evaluation, and some parameters of the specimen can only be obtained by these means, such as limit conditions to maintain serviceability (e.g. yield point, tensile strength, fatigue, impact resistance, melting point, isolation voltage, maximum power dissipation) [7]. Given that the tested specimen is a legitimate representative of a set of materials or specimens, destructive tests can result in standards and specification for the kind of tested material.

Table 2.1: NDE methods

Method	Principles	Application	Advantages	Limitations
Visual	Uses reflected or transmitted light from test object that is imaged with the human eye or other light-sensing device	Many applications in many industries ranging from raw material to finished products and in-service inspection	Can be inexpensive and simple with minimal training required. Broad scope of uses and benefits	Only surface conditions can be evaluated. Effective source of illumination required. Access necessary
Penetrant liquid	A liquid containing visible or fluorescent dye is applied to surface and enters discontinuities by capillary action	Virtually any solid nonabsorbent material having uncoated surfaces that are not contaminated	Relatively easy and materials are inexpensive. Extremely sensitive, very versatile. Minimal training	Discontinuities open to the surface only. Surface conditions must be relatively smooth and free of contaminants
Magnetic particle	Test part is magnetized and fine ferromagnetic particles applied to surface, aligning at discontinuity	All ferromagnetic materials, for surface and slightly subsurface discontinuities; large and small parts	Relatively easy to use. Equipment/material usually inexpensive. Highly sensitive and fast compared to penetrant liquid	Only surface and a few subsurface discontinuities can be detected. Ferromagnetic materials only
Radiographic	Radiographic film is exposed when radiation passes through the test object. Discontinuities affect exposure	Most materials, shapes and structures. Examples include welds, castings, composites, etc., as manufactured or in-service	Provides a permanent record and high sensitivity. Most widely used and accepted volumetric examination	Limited thickness based on material density.Orientation of planar discontinuities is critical. Radiation hazard
Ultrasonic	High-frequency sound pulses from a transducer propagate through the test material, reflecting at interfaces	Most materials can be examined if sound transmission and surface finish are good and shape is not complex	Provides precise, high-sensitivity results quickly. Thickness information, depth, and type of flaw can be obtained from one side of the component	No permanent record (usually). Material attenuation, surface finish and contour. Requires couplant
Eddy current	Localized electrical fields are induced into a conductive test specimen by electromagnetic induction	Virtually all conductive materials can be examined for flaws, metallurgical conditions, thinning, and conductivity	Quick, versatile, sensitive; can be noncontacting; easily adaptable to automation an in-situ examinations	Variables must be understood and controlled. Shallow-depth of penetration, lift-off effects and surface condition

2.2 Welding and NDE

The process of welding consists in bringing two materials together and cause them to join [21], with goal of achieving the materials' union with no discontinuities either by using only the involved specimens or by using a filling material between them. This concept paved the technological progress of fabrication processes and still plays a major role in modern industry [21].

In order to achieve continuity in materials bonding, the welding process seeks to establish homogeneous distribution in the atomic structure between the material's interfaces, producing crystalline patterns that conserve at least the original mechanical properties of the once separated pieces. Such process requires at least one of the following physical quantities:

- **Temperature**. Increasing the temperature (heating) enables the diffusion of atoms through the material's structure, by either melting the specimens and joining them by fusion or softening the material in order to enable atomic diffusion through its structure. Moreover, the process also drives off volatile contaminants present at the material's composition.

- **Pressure**. Applying pressure at the interface results in plastic deformation of asperity at the materials' surface, thus bringing the atoms close to homogeneous arrangement. It also has the effect of breaking layers or adsorbed gases and moisture

Most practical welding methods employ a great amount of heating and low pressure levels [21]. When it comes to fusion, three major methods are employed [22]:

- Gas welding, when fusion is driven by a flame caused by the reaction between a fuel gas and oxygen;

- Arc welding, when the fusion is carried out with an electric arc between the material and an electrode;

- High energy beam welding, when the fusion is caused by an electron or laser beam.

This work will evaluate weld beads obtained as a result of arc welding process. In order to produce the electrical arc, an electrode is used in addition to inert gas shielding, whose purpose is to keep the surface free of oxidation while the welding process takes place. The electrode can be consumable, meaning that it provides the filler material, or non-consumable, where the filler material must be externally sourced. A general representation of arc welding process is depicted in Figure 2.1.

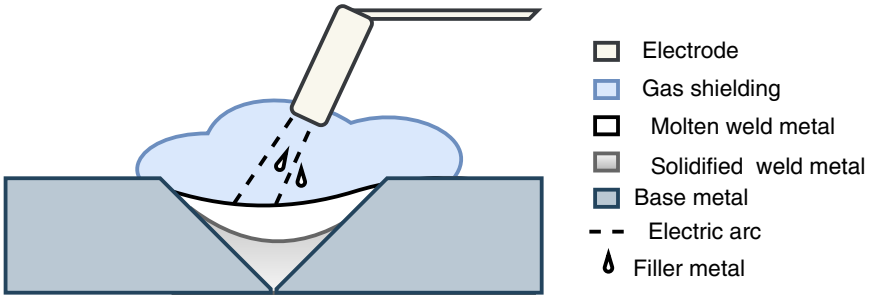


Figure 2.1: Representation of arc welding process.

The Gas-Metal Arc Welding (GMAW) uses a consumable electrode as the electric arc producer whereas in the Gas-Tungsten Arc Welding (GTAW) method, a non-consumable tungsten electrode is used do produce the electric arc. Both methods have the welding process shielded by inert gases such as helium and argon, which access the melting pool by outlets in the electrode. In Shielded Metal Arc Welding (SMAW), a consumable electrode is covered with a flux layer, in a manner that the heat caused by the electric arc produces gas from the covering flux.

Since welding is employed to join materials, continuity in the weld region must be granted. Discontinuities present at the weld may reduce mechanical resistance and cause failure while the specimen is under service. The main discontinuities that may occur in welds are shown in Table 2.2. Some of the occurrences in this Table are related to discontinuities in the form of cracks inside the weld region or at the interface between the joined materials and the welding (for instance in undercut, lack of fusion and lack of penetration), while other flaws are caused by trapped material (either gas in the case of porosity or different material originated in the welding process in the case of slag inclusion).

The nature of the discontinuities shown in Table 2.2 present properties that make many NDE methods shown in Section 2.1 suitable for weld evaluation. Discontinuities may alter induced magnetic fields and induced currents in metallic workpieces'

Table 2.2: Common discontinuities in welding processes

Discontinuity	Description	Causes
Slag Inclusion	Presence of material other than base and filler	Welding processes that use protective cover (e.g. SMAW)
Cracks	Air-filled gaps inside the weld	Unrestrained expansion of the base material during melting. After solidification, restraints on contraction of the base material may cause forces that originate cracks and fractures.
Porosity	Gaps caused by gas trapped in weld that dissipate after solidification	Use of gas during melting
Lack of Fusion	Gap between base and filler material	Incomplete melting of a material during welding process
Undercut	Ditching formation at the edge of the fusion zone on the surface of base material	Improper welding parameters such as too high rate of arc motion along the workpiece (travel speed)
Lack of Penetration	Incomplete joint filling	Incorrect electrode angle and incorrect electrode current

surfaces and block x-ray incidence on radiographic films while cracks may accumulate penetrant liquid and reflect ultrasound waves. Therefore, ultrasound, radiographic, liquid penetrant, magnetic particle and eddy current NDE methods find great usage in weld evaluation [7].

2.3 Fundamentals of Ultrasonic NDE

Ultrasonic NDE is carried out by generating a mechanical wave with frequency in the MHz range (0.5 MHz to 25 MHz [7]) and introducing it into a solid material, and then analyzing time delays between emitted and received waves and also their waveforms. Different properties of the material will produce different types of travelling elastic waves, each type carrying their own "signature", meaning that by analyzing the received signal, it is possible to retrieve information about what is the actual physical condition of the studied region in the material.

While characterizing a setup for ultrasonic NDE, one must pay attention to certain aspects involving mechanical waves propagating in media, such as wavelength, speed of the travelling wave in the medium and density of the medium. The way by which inspection of inner conditions of the material happens is analyzing the reflection and diffraction waves at the receiver probe. However, these wave phenomena are strongly related to the size of the obstacles faced by the waves. Obstacles must have at least the major dimension in the range of half wavelength in order to produce reflections [1] so this constraint, along with the wave speed in the medium, must determine the frequency of the signal.

Reflection at obstacles inside a specimen happen due to different densities that may occur between different media regions. The specific acoustic impedance Z of a material is defined as the ratio of acoustic pressure p per particle velocity U in the material:

$$Z = \frac{p}{U} \tag{2.1}$$

In case of non-attenuating progressing plane waves, this number has no imaginary component [23] and can be obtained with Equation 2.2:

$$Z = \rho V \tag{2.2}$$

where ρ is the density of the material and V is the speed of the wave in the material. Typical units for Z, ρ and V are kg/m^{2}s, kg/m^3 and m/s respectively.

The ratio of energy reflection between two different materials is given by [1]

$$R = \left(\frac{Z_1 - Z_2}{Z_1 + Z_2} \right)^2 \times 100\% \tag{2.3}$$

where Z_1 and Z_2 are the acoustic impedances of the materials surrounding the interface (Figure 2.2).

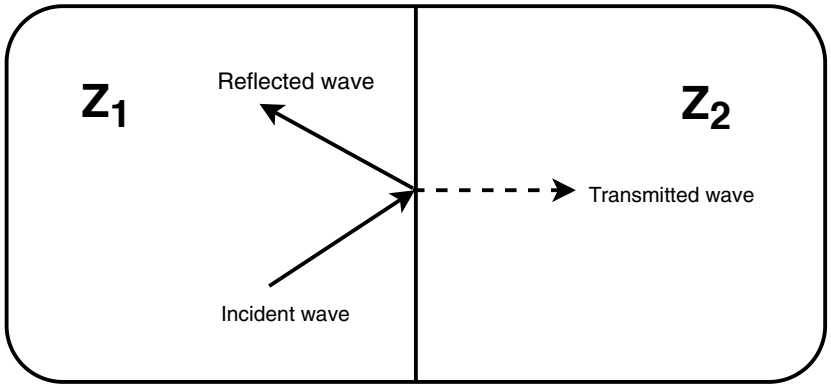


Figure 2.2: Wave behaviour in an interface. Reflection and transmission angles may vary according to the material impedances.

In the case of cracks or other obstacles filled with air or other less dense material inside steel plates, the reflected energy ratio is almost 100% (steel and air have acoustic impedance of 46.7 and 0.004 respectively). This phenomenon keeps ultrasonic beams trapped inside the specimen and provides strong reflection signals to be detected at receptor probes, which carry evidence of discontinuities.

The downside of the ratio of reflected energy between high and low density materials is the fact that in order to deploy the ultrasonic beam into the specimen, a thin layer of air will almost always be present, meaning that the energy transfer to the material would be impractical. One method to eliminate the air in the process is to fill in the gap between the probe and the specimen with a substance with higher acoustic impedance to increase the ratio of transferred energy. Such substances are known as couplants, and generally are found in the form of gels, pastes or liquids. Immersion tests, where the procedure takes place inside a liquid media, are also an alternative for the problem of acoustic wave transmission between the emitter probe and specimen. Most couplants permit a ratio of 10 to 15% of transmission.

Regarding the probes, some types are employed in ultrasound NDE [1]:

- **Piezoelectric**. This is a characteristic of materials that generate voltage when submitted to mechanical distortion (compression and expansion), with the opposite effect also valid (applying voltage produces mechanical distortions). This kind of probe generally has polycrystalline ceramic crystals in its composition and it is the most used component for probes.

- **EMAT**. EMAT stands for ElectroMagnetic Acoustic Transducer. It is a non contact method (therefore it doesn't need couplant) that produces mechanical waves in the specimen by deformations caused by magnetic interaction between the specimen and the electromagnetic field.

- **Laser-Generated ultrasound**. This method uses a laser probe to produce ultrasonic waves at the probe through the use of cycles of heating, which causes momentaneous thermal expansion of the specimen's material.

Piezoelectric probes are the most employed probe in NDE. Commercial models found in market can demand voltage levels ranging from 100 to 1000 V spike excitation while dissipating up to 125 mW of power, thus requiring special electronic conditioning to drive the transducer [24]. At the reception probe, the signals create oscillations at voltage levels ranging from 0.001 to 1 V, therefore requiring amplification in order to be displayed at an instrument.

The probe deploys the signal into the specimen but since it has measurable physical dimensions, it cannot be considered a punctual source of emission, but instead a finite source. Finite sources can be viewed as a collection of point sources, which means that it may occur wave interference (constructive and destructive) between these sources in a region near them. Only when measured at certain distance, these effects cease to exist and the wave beam can be considered as emitted by a point source and treated as a straight wave beam. The region where interference effects occur and measures must be avoided is called near field, and the near field distance NF to the emitting source is obtained by [1]

$$NF = \frac{D^2}{4\lambda}, \tag{2.4}$$

where λ is the wavelength of the emitted wave and D is the probe diameter. Alternatively,

$$NF = \frac{D^2 f}{4V}. \tag{2.5}$$

From the properties of the ultrasonic NDE explained in this chapter, it can be seen that the results from this technique are in general more difficult to interpret than other non-destructive methods. The information derived from ultrasonic NDE results are deduced from interpretation of a time domain series of pulsed waveforms, which can be associated to several physical conditions in the specimen that can be as trivial as the specimen's geometric boundaries or as tricky as reflection and/or diffraction from discontinuities or crack tips, for instance.

These factors make the method heavily dependent on the knowledge and experience of the inspector in charge of the evaluation, where he or she must correctly identify within the time window of the resulting signal which waveforms are correspondent to the geometric boundaries and which ones are caused by discontinuities. It may also be important to identify the kind of discontinuity. Therefore, the evaluation is prone to human failure.

This paradigm reinforces the suitability of a decision support system as mentioned in section 1.1, in a manner that the inspector can be assisted by a system that works in consonance with the inspector's ability while diminishing the influence of human failure.

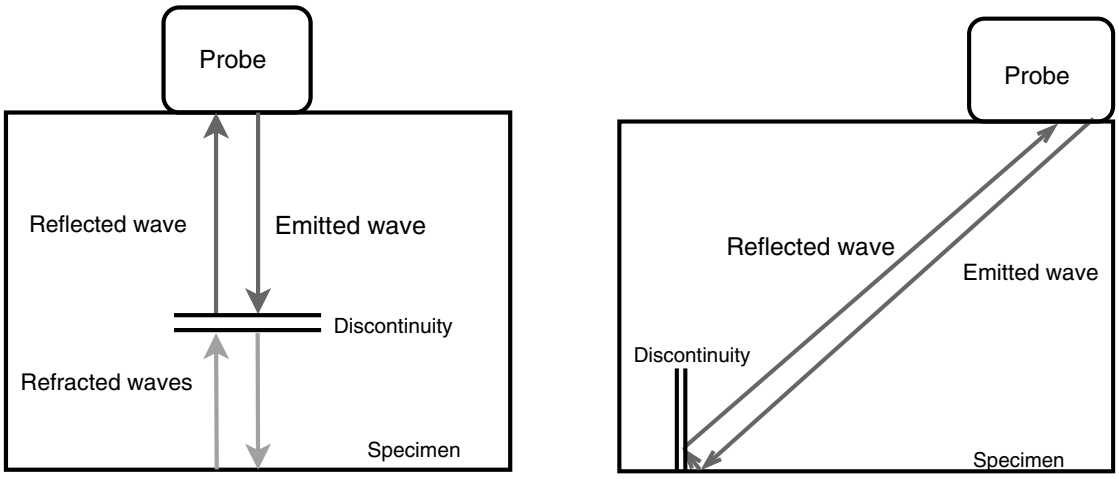
Ultrasonic NDE is suitable for inspection in weld beads. In many cases, the welding represents a critical point in the structure of the specimen. As seen in Section 4.2, many types of flaws can occur on weld beads including cracks, incomplete welding process, presence of outer solid particles, etc. In the next sections, some methods of ultrasonic NDE will be presented.

2.4 Pulse-Echo

One of the most popular ultrasound NDE method, the pulse-echo evaluation detects flaws by measuring the echo response of an emitted signal at the direction of the assumed flaw inside the specimen. It is a pulse method, which means that it is based on the observation of travelling short waves [23], and it consists in emitting a pulse

of ultrasonic wave and wait for its return (the echo). This method is simple and is advantageous on specimens where access may exist only from one side.

The received echo can be either due to backwall reflection (when the pulse traverses the entire thickness of the material) or due to reflections on discontinuities present inside of the material, as shown in Figures 2.3a and 2.3b. In this way, it can be possible to inspect and size flaws by analyzing time intervals in the echo signal (Figure 2.4).



(a) Discontinuity inside the specimen (b) Discontinuity at back surface

Figure 2.3: Pulse-echo technique.

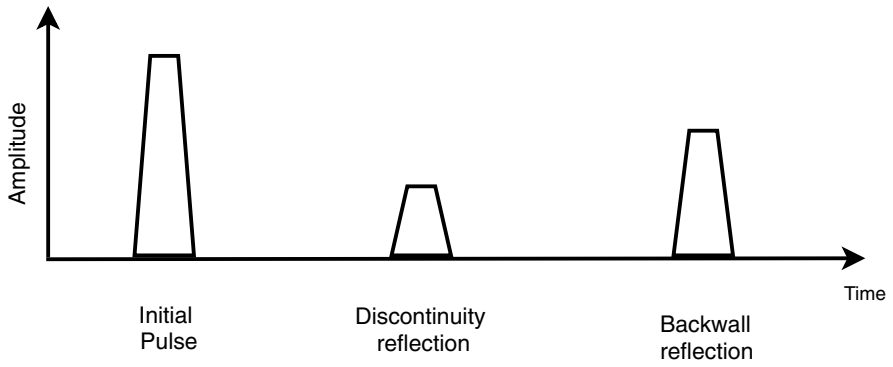


Figure 2.4: Representation of Pulse-echo signal

2.5 Time-of-Flight Diffraction

Pulse-echo method is limited to certain configurations of flaws [25], because it is assumed that planar features suitable angled inside the specimen will produce echoes.

Thus, newer techniques based on diffraction waves such as Time-of-Flight Diffraction (ToFD) rose to overcome the issues in Pulse-Echo technique. ToFD uses a pair of probes laid out in a specific arrangement [25]. An emitter and a receiver probes are placed along the surface of the material at a usually 45° inclination angle, each one on opposite sides relative to the area of interest to be investigated.

Once the signal beam is deployed into the surface of the material, the following phenomena will occur, depicted together in Figure 2.5:

- A lateral wave will propagate through the surface of the material, being the first signal to arrive at the sensor probe.
- Further signals can excite the sensor probe, due to diffraction originated from internal obstacles present in the material.
- The last signal received at the sensor probe is the one caused by reflection on the back surface of the material, the backwall reflection wave.

The profile of the signal containing the received waves (Figure 2.6) is the object of study in this method, since the distance between each other along with their waveform carry information about the inner configuration of the material.

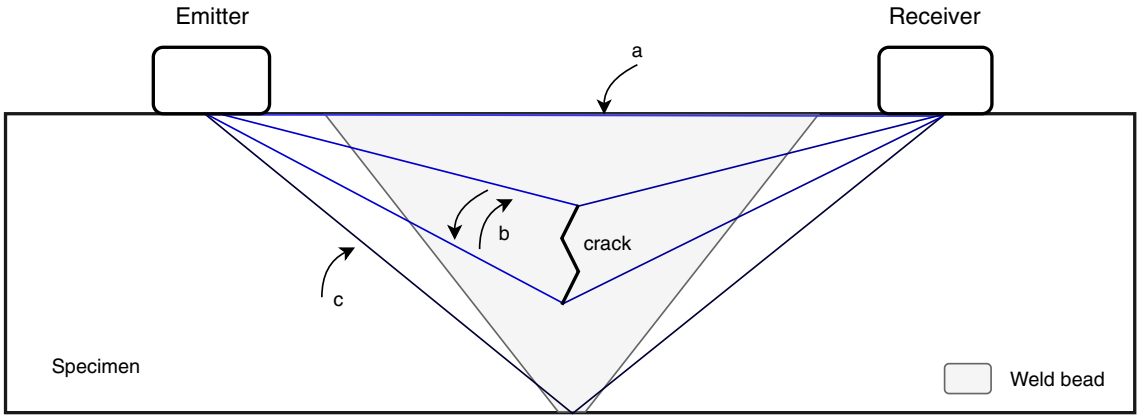


Figure 2.5: Wave paths in ToFD technique. **a**: Lateral wave. **b**: Diffraction waves. **c**: Backwall wave.

2.6 Other ultrasound NDE methods

Despite the broad use of Pulse-echo and ToFD methods in ultrasonic NDE, other methods are employed in this category of non-destructive evaluation. Following are

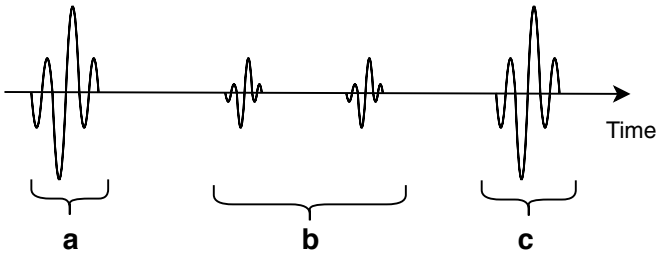


Figure 2.6: Representation of the received ToFD signal in Figure 2.5.

some examples.

2.6.1 Through-transmission

This method is also known as shadow method, and it places two probes symmetrically at opposite sides of a test specimen. A continuous wave is emitted by one probe and received by the other. This test is carried out on specimens whose material produce strong attenuation on mechanical waves so that detection of pulse reflections cannot happen [1].

At the receiver probe, the signal has its amplitude compared to a standard measure, and reduction of the amplitude indicates that likely an obstacle is covering the signal. Since it is based on amplitude values, this method is sensible to couplant losses. Figure 2.7 depicts the method usage.

(a) Discontinuity-free reception. (b) Reception blocked by discontinuity.

Figure 2.7: Through transmission method.

2.6.2 Lamb wave technique

Lamb waves are surface waves that causes particles to move in an elliptical shape within the specimen [1, 23]. When producing Lamb waves, the probes are placed at the surface of the specimen. This waves occur in specimens where their thickness are

a few multiples of the exciting signal's wavelength, meaning that this phenomenon is restricted to plates, wires and tubes. These waves are sensitive to surface flaws and discontinuities, and since they suffer low attenuation, tests with lamb waves are carried out to provide fast inspection in big structures. Figure 2.8 shows a representation of NDE using Lamb waves.

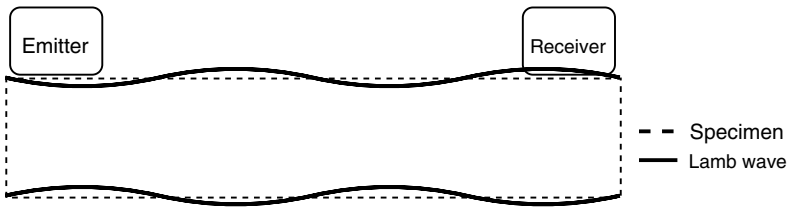


Figure 2.8: Evaluation using Lamb waves.

2.6.3 Phased arrays

Mechanically executed tests, where the probes are displaced along the specimen by the inspectors, are slow and thus may not fulfill the requirements for inspection in a real-time basis. One method to overcome this fact is by using a series of probes displaced along the evaluated specimen, whose pulse beams are controlled time-wise in an automatic fashion. This way, a great amount of data can be stored in short time. Combining different beam phases can also produce desired beam shapes at variable depths [1].

This method requires much more material and additional computational system available.

2.7 Ultrasound NDE and Decision Support Systems in literature

Investigation on flaw classification using ultrasonic signals and artificial intelligence has been of interest in many works on the literature. In [26], Veiga *et al.*, Pulse-echo and ToFD signals are used to train artificial neural network-based classifiers for 4 types of defects in weld beads. Seyedtabaii [27] evaluates the accuracy of several types of neural classifiers regarding four types of flaws, with time domain and

frequency features from Pulse-echo signals, with an average test accuracy of 75%. Also with pulse-echo signals, Liu *et al.* [28] analyze an artificial neural network-based classifier for 4 classes of flaws in resistance spot weld, using time, frequency and wavelet domain features. Pulse-echo technique is also used by Kesharaju and Nagarajah [4] on silicon carbide ceramics with a five-class artificial neural network classifier. Crack sizing is evaluated by Her and Lin [5] using pulse-echo signal in frequency domain to train a classifier. In [29], Simas Filho *et al.* use pulse-echo signal in frequency domain to train a neural network-based defect classifier for fiber metal laminates, with frequency domain feature preprocessed in order to extract fewer significant features.

ToFD is used in Murta *et al.* work [30] where, in a simulated 2D environment, the time domain signals are preprocessed and submitted to classification based in clustering algorithms for 3 classes of defect. The overall success rate is 76.85%. In [2], Carvalho *et al.* use ToFD time domain signal preprocessed with polynomial filter to train neural network classifier, with average success rate of 75.13% and maximum success rate of 86%. The ToFD technique is employed in [31] with an array of sensors to support an algorithm that estimates size and position of flaws inside a specimen.

As seen in the works mentioned above, Artificial Neural Networks is a popular method to build decision support systems in ultrasonic NDE. However, the reported neural networks use the traditional method for the training step (*(*backpropagation), which is not suitable for the embedded platform intended to be used in this thesis.

2.8 Final Remarks

This chapter presented the theoretical background and practical aspects for non-destructive evaluation, focusing on ultrasound-based NDE. Several methods for ultrasonic NDE were presented along with the technique used in this work, Time-of-flight Diffraction. Moreover, welding processes were also presented, showing a few types of defects that may result from some processes and how NDE is a suitable technique for identifying them. Finally, it was shown works in the literature that assess ultrasound NDE with the aid of decision support systems in personal com-

puter environments. The design of an embedded decision support system for weld defect classification will be discussed in the next chapters.

Chapter 3

Artificial Neural Networks and Embedded Systems

3.1 Introduction

This chapter presents an overview on traditional neural classifiers and embedded systems. Aspects of classification and feature extraction methods used in this thesis (Discrete Fourier Transform, Principal Component Analysis and Autoencoders) are presented. The formulation of the particular classifiers evaluated in this work are shown, namely backpropagation-trained Artificial Neural Networks and Extreme Learning Machines. The last Section in this chapter presents core concepts of microcontrollers and a review on neural networks implementation on such devices.

3.2 Pattern Recognition and Classification

In a problem where the acquired data origin must be tracked down to one out of a certain number of sources, the task of classification takes place. The so called classifier must assign an actual input presented to the system's sensors into one of the known classes [32]. The system maps the input's features into a new space of possibly different dimensionality in which a decision rule operates. In order to work properly, such system must be efficient in the task of classification (minimize assignment errors) and have convenient execution time within the system's processing chain.

The simplest case of decision is the one for binary classification [33]. In this

problem, a certain sample must be classified into one out of two classes: the positive class (or positive hypothesis) H_1, which is the class of interest, and the negative class (or negative hypothesis) H_0. In a binary classification problem, the result might lead to one out of four scenarios:

- H_1 is assigned to the input, when it belongs to H_1 (true positive);
- H_1 is assigned to the input, when it belongs to H_0 (false positive);
- H_0 is assigned to the input, when it belongs to H_1 (false negative);
- H_0 is assigned to the input, when it belongs to H_0 (true negative).

Depending on the problem, a high rate of false positives is undesirable: in the present work, a compromised structure being evaluated as defect-free might lead to interruption of service and even human life losses. For other problems, false negatives must present the lowest possible rate, as it is the case in medical diagnosis, where diseases and bad health conditions are part of the class of interest.

The problem of binary classification can extend to the multiclass case. In this problem, M classes can be assigned to a given input. The classifier in this case also maps the input's feature into a new space, but with more decision rules to separate the classes. The problem of M classes can be tackled by decomposing it into cascaded binary classifiers, or using a proper discriminator to the multiclass problem.

In the case of the multiclass problem, it is usual to represent the results of the classifier after a certain number of classifications in the form of confusion matrix (Table 3.1). The confusion matrix [34] is a $M \times M$ matrix where each row represents a reference class, and each column represents the assigned class. Therefore, the main diagonal (elements $m_{ij}, i = j$ in Table 3.1) contains the amount of correctly assigned inputs, either in absolute or percentage representation and the remaining elements $m_{ij}, i \neq j$ represent misclassification errors (an element from class i being assigned as class j). Thus, the more the values spread in a given row, the lower the classification performance for that specific class.

Table 3.1: Representation of a confusion matrix

	Class 1	Class 2	...	Class M
Class 1	m_{11}	m_{12}		m_{1M}
Class 2	m_{21}	m_{22}		m_{2M}
$\vdots$			$\ddots$	
Class M	m_{M1}	m_{M2}		m_{MM}

3.3 Feature Selection and Extraction

Systems that behave as classifiers map their inputs onto a new space, as mentioned in Section 3.2. This mapping works on characteristics that the classifier "sees" once a sample is presented at its inputs.

When data has naturally low dimensionality, such as 3-D coordinates of a point in the space, feature extraction is not a concern while designing a classifier. However, as input dimensionality increases, such as signals comprising time samples, some method of dimensionality reduction must be used in order to reduce the classifier complexity. Moreover, some applications where data storage is a tight constraint, dimensionality reduction is mandatory. Two common approaches for dimensionality reduction are feature selection and feature extraction [32].

Feature selection is the process of selecting the l features present in a sample with d total features that provide most valuable information to a classifier in order to achieve a good performance. The other $(d - l)$ features are then discarded, and the final sample dimension is lower than the original.

Depending on the nature of the sample, the feature selection approach may be too complex. In this case, the method referred to as feature extraction consists of transforming the sample into a new feature space, where in this new space there are $t < d$ features, which are mapped from the original d features. Due to the nature of the data used in this work (see Section 4.5), feature extraction methods will be considered.

Some methods commonly used for feature extraction are listed below:

- **Discrete Fourier Transform (DFT)**. The DFT (Equation 3.1) [35] is a mathematical operation that converts a discrete signal of length L usually in

time domain to the frequency domain. It is an advantageous tool to extract frequency information from stationary signals.

$$X[m] = \sum_{n=0}^{L-1} x[n] e^{-j2\pi \frac{mn}{L}}, m = 0, 1, \ldots, L-1 \tag{3.1}$$

For $x[n]$ real-valued, which is always the case for acquired signals, $|X[m]|$ is even, meaning that the resulting dimensionality is halved when considering magnitude data. Moreover, certain regions at the discrete frequency spectrum have low energy content, so that they can be neglected as inputs to the classifier.

- **Principal Component Analysis (PCA)**.

  Principal Component Analysis is a method for feature extraction that aims to represent data with accuracy while reducing dimensionality [32]. It's a linear transformation $\mathbf{T}$, achieved through diagonalization of the covariance matrix of the original data set. Each component in this new representation (Equation 3.2) is a linear combination of the original variables, being organized in descending order of variance. Therefore, the first component represents the direction of most variance in the data set, the second component accounts for the direction of second most variance, and so on. Moreover, each component is orthogonal to the previous ones. In some cases, observing the components of greater variance may be a good measure to extract discriminant features.

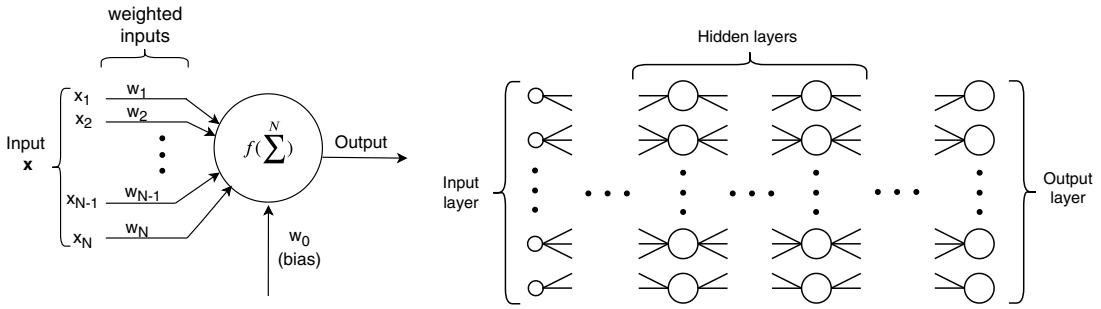
$$\mathbf{P} = \mathbf{TX} \tag{3.2}$$

  After transforming the dataset $\mathbf{X}$ using $\mathbf{T}$, the new data representation $\mathbf{P}$ has the same dimensionality as the original data $\mathbf{X}_{n \times N}$. However, since its new components are presented in a descending order of contribution to the total variance in the data set, usually one can rely on the first principal components to describe the studied data, since the last components have little contribution to variance and can be discarded for data compaction purposes [32], which results in $\mathbf{P}_{r \times N}, r < n$. The consequence is dimensionality reduction and less complexity to classification systems.

3.4 Artificial Neural Networks and Backpropagation Algorithm

Artificial Neural Network (ANN) is a computational method inspired on a network of biological cells called neurons and the way they are connected [12]. In living beings, such network comprises numerous neurons interconnected that act as pathways to incoming data which provide the organic responses and behavior to environmental stimulation. Following the same philosophy, the artificial neural network is intended to use a network of interconnected abstract elements that produce desired responses to a given external stimulation.

The neuron on an ANN is a mathematical entity that connects to an input, another neuron or an output by weighted signal paths. The mathematical model for a neuron of the most popular type of ANN, the Multilayer Perceptron (MLP), is shown in 3.1a and the general topology of an ANN is shown in Figure3.1b.



(a) Mathematical model for a neuron (perceptron).

(b) General topology of an ANN.

Figure 3.1: Illustration of parts of an Artificial Neural Network.

As shown in Figure 3.1b, the Artificial Neural Network comprises:

- an input layer, which contains the input nodes (the sources for the inputs to the network);
- a certain number of hidden layers, each one with their own neuron count;
- an output layer with neurons that provide the network output.

The network can have all of its neurons interconnected between two consecutive layers (full connected), meaning that every neuron in one layer has connections to

each neuron in the previous layer, and also can have connections linking neurons in a layer to neurons in back layers (feedback connections). In this work, the evaluated networks will be of the type full connected *feedforward* network, meaning that the signal is propagated only in forward direction (from input to output layers), with no feedback connections.

The output of an artificial neuron is numerically evaluated in two steps. Firstly, the weighted inputs are summed. Considering I elements in input vector $\mathbf{x}$ with respective weights in vector $\mathbf{w}$, it will result in

$$y = \mathbf{w}^T\mathbf{x} + b \tag{3.3}$$

wherein b is often called *bias* to the neuron and acts as an arbitrary offset to the inner product between $\mathbf{w}$ and $\mathbf{x}$.

This amount then is used as argument for an activation function $f(\cdot)$. The output of the neuron is in the form

$$h = f(y) = f(\mathbf{w}^T\mathbf{x} + b) \tag{3.4}$$

It is usual to have the activation function with a limiter shape, such as the hard limiter (Equation 3.5), the Rectified Linear Unit (ReLU, Equation 3.6) and the Sigmoid (Equation 3.7), a differentiable non-linear limiter.

$$L(x) = \begin{cases} -1 & for \quad x < 0, \\ 1 & for \quad x \geq 0 \end{cases} \tag{3.5}$$

$$R(x) = \begin{cases} 0 & for \quad x < 0, \\ x & for \quad x \geq 0 \end{cases} \tag{3.6}$$

$$S(x) = \frac{1}{1 + e^{-ax}}, \quad a \in \mathbb{R}_{>0} \tag{3.7}$$

The activation function is often used as a way to control the neuron outputs, and must be chosen according to the desired behaviour of the network. ANNs are either usually employed in regression problems, where it is desired that a given input produces a numeric output, or classification problems, where a given input must be categorized into specific classes. When trying to build a model for regression, certain neurons, specially the ones at the output layer, should have activation functions

with large range of values. When training classifiers, it is desirable to have the neurons behaving with functions that categorize data, which is the case of limiters. This way, the neuron will be "activated" if the internal product between input and weight vectors reaches a critical level.

In order to have an ANN to be suitable for a problem, a training step must occur prior to using it. ANNs are a type of artificial intelligence that may use supervised training, meaning that pairs of input-output are provided during the training step so that the network will "learn" by assigning the provided target outputs to their respective samples and the trained model will be able to generalize outputs for test inputs unknown to the network. Therefore, available samples from a given subject usually are divided into training, validation (to avoid overfitting[1] in some training methods) and test sets [32]. In this work, only networks with supervised training are going to be considered.

The traditional method to train ANNs is based on minimizing the mean square error, using a gradient descent algorithm, between the current output for a given sample and its desired target by adjusting the weights (connections) of neurons. This method is known as *backpropagation*, since it starts at the output neurons backwards to the neurons at inner layers. The error at the output of a neuron at the output layer is

$$e(k) = d(k) - h(k) \tag{3.8}$$

where $d(k)$ is the desired output and $h(k)$ is the current output of the neuron according to Equation 3.4 an the index k denotes the current iteration in the training method. The backpropagation method seeks to minimize the average error energy for each neuron at the output layer for the whole sample set used during the training procedure. The instantaneous quadratic error energy for a neuron j in the output layer is

$$E_j(k) = \frac{1}{2} e_j^2(k) \tag{3.9}$$

[1] Overfitting occurs when the network looses generalization capability and cannot produce good outputs except for the ones of the training set [36].

and the accumulated error energy for the output layer with m neurons is

$$E_{acc}(k) = \frac{1}{2}\sum_{j=1}^{m} e_j^2(k) \tag{3.10}$$

and the average error energy for a sample set with N samples is

$$E_{avg} = \frac{1}{N}\sum_{k=1}^{N} E_{acc}(k) \tag{3.11}$$

The weight adjustment $\Delta\mathbf{w}(k)$ in the backpropagation algorithm follows the same pattern of the Least Mean Square (LMS) algorithm [12], which is updating the weight values in a direction opposite to the gradient of the error energy with respect to $\mathbf{w}$:

$$\Delta\mathbf{w}(k) = -\eta\frac{\partial E(k)}{\partial\mathbf{w}(k)} \tag{3.12}$$

where η is called learning rate, and dictates the pace in which the weight adjustment lead to minimum error energy. Common values for η lie in the interval $0.1 \leq \eta \leq 0.4$ [32].

Following the chain rule [12],

$$\frac{\partial E(k)}{\partial\mathbf{w}(k)} = \frac{\partial E(k)}{\partial e(k)}\frac{\partial e(k)}{\partial h(k)}\frac{\partial h(k)}{\partial y(k)}\frac{\partial y(k)}{\partial\mathbf{w}(k)} \tag{3.13}$$

Applying it to equations 3.9, 3.8, 3.4 and 3.3:

$$\frac{\partial E(k)}{\partial e(k)} = e(k) \tag{3.14}$$

$$\frac{\partial e(k)}{\partial h(k)} = -1 \tag{3.15}$$

$$\frac{\partial h(k)}{\partial f(k)} = f'(y(k)) \tag{3.16}$$

$$\frac{\partial y(k)}{\partial\mathbf{w}(k)} = \mathbf{x}(k) \tag{3.17}$$

where $f'(\cdot)$ denotes the first derivative with respect to the function's argument. Therefore,

$$\frac{\partial E(k)}{\partial \mathbf{w}(k)} = -e(k)f'(y(k))\mathbf{x}(k) \tag{3.18}$$

and

$$\Delta\mathbf{w}(k) = \eta e(k)f'(y(k))\mathbf{x}(k) \tag{3.19}$$

Rewriting Equation 3.19 leads to

$$\Delta\mathbf{w}(k+1) = \eta e(k)f'(y(k))\mathbf{x}(k) + \mathbf{w}(k) \tag{3.20}$$

The term $e(k)f'(y(k))$ is called local gradient $\delta(k)$ of the neuron.

Equation 3.20 shows that in order to adjust the output weights, the desired output (implicit in the error $e(k)$), the inputs from the previous layer and the first derivative of the activation function are required. It also shows that the method requires a differentiable activation function, and this is the reason why the sigmoid function is preferred when a limiter function must be chosen.

The desired output is a known value for the neurons at the output layer. However, for the neurons at hidden layers, there is no such value. Therefore, when adjusting the weights in a hidden layer, a slightly different approach must be taken, where the error of a hidden neuron is influenced by the error of all neurons to which it is connected in the layer immediately to the right. The weight adjustment in a hidden neuron in layer l with connections to a layer r is obtained by [12]

$$\begin{aligned} \Delta\mathbf{w}_l(k+1) &= \eta\delta_l(k)\mathbf{x}(k) + \mathbf{w}(k) \\ \delta(k) &= f'(y(k)) + \textstyle\sum_r \delta_r(k)w_{lr} \end{aligned} \tag{3.21}$$

with w_{lr} being the weight connecting a neuron in layer l to layer r.

The training procedure for backpropagation can now be formulated as

1. Generate output by presenting inputs to the input layer;

2. Start the backpropagation procedure in each neuron at the output layer. In this step, the error $e(k)$ feeds the local gradient (Equation 3.20);

3. Proceed to the left layer and update the weights in each neuron using Equation 3.21 and continue this procedure until reach the first hidden layer.

The described procedure is executed for the N samples in the training set, then again until the change in mean squared error energy reaches an accepted threshold. The backpropagation procedure demands a great deal of calculations in order to obtain derivatives at each neuron. Moreover, the learning rate can play a crucial role in the algorithm convergence, as lower values can lead to slower performances, whereas larger values can make the error oscillate around the desired minimum.

Due to being an established method in the field of artificial intelligence, ANNs trained with backpropagation have been disseminated among many fields in science and technology. Recent works studying face orientation recognition [37], prediction of monsoon rainfall [38], postharvest banana quality classification [39], prediction of shear resistance in concrete beams [40], prediction of finish cooling temperature in thermal control processes [41], production of watermarks in online security [42] and tuning of PID controllers [43] are examples of the variance in fields of knowledge that still rely on backpropagation ANNs to carry out research.

3.4.1 Modifications to the backpropagation algorithm

Modifications to the backpropagation procedure have been proposed since its inception, in an attempt to either make the method faster, or solve convergence issues. In this work, the Levenberg-Marquardt and RPROP methods will be evaluated for their training speed and classification efficiency against a different class of neural networks called Extreme Learning Machines. These concurrent methods were chosen for their faster performance in the training step compared to the traditional backpropagation algorithm. They are detailed below:

- **Levenberg-Marquardt Algorithm (LM)**. This method, developed by Kenneth Levenberg [44] and later improved by Donald Marquardt [45] intends to solve the problem of minimization of the sum of least squares of non-linear functions. It combines steep descent methods based on gradient with the Gauss-Newton method, which is used to minimize a sum of squares [46]. It uses an adaptive parameter that changes the behaviour of the algorithm according to the distance to the minimum error. This way, the weights adjustment uses the advantage of fast convergence of Gauss-Newton method with the stability of the steep descent method, generally converging faster than the

original backpropagation training [47].

- **Resilient Backpropagation (RPROP)**. RPROP [48] is a gradient-search algorithm that doesn't require to compute the magnitude of the local derivatives in weight adjustment. The goal in this method is to search for signal variations in derivatives of the error function, which implies in crossing a zero value. Since the target for the minimization problem is a nearly equal to zero derivative, the learning factor η, by which the weights are adjusted, is set accordingly to the change or maintenance of signal. This improves the speed of the algorithm, since many calculations can be left aside.

3.5 Extreme Learning Machines

Extreme Learning Machine (ELM) [49] is a theory in the field of Neural Networks with Random Weights [50] for building and training ANNs in the Single Feed-Forward Network (SLFN) topology. This means that the network has only one hidden layer and no feedback connections. ELMs are tuned for a faster training step (regarding execution time) and similar accuracy to established training methods based on gradient search optimization.

An ELM has its hidden neurons initialized with fixed random weights connecting the input layer to the hidden layer prior to its training. The goal is to determine the weights of the neurons connecting the hidden layer to the output layer.

In order to train an ELM ANN with n nodes at the input layer, L neurons at the hidden layer and m neurons at the output layer, there must be N samples available in input-output pairs $(\mathbf{x_i}, \mathbf{t_i})$, $\mathbf{x_i} \in \mathbb{R}^n$, $\mathbf{t_i} \in \mathbb{R}^m$, $i = 1, 2 \ldots N$. The ELM is a network with $\mathbf{w}_j \in \mathbb{R}^n$ random weights associated with b_j bias, $j = 1, 2 \ldots L$ defining the neurons connecting the input layer to the output layer. The neuron output in ELM is similar to the one described in equation 3.4. Finally, $\boldsymbol{\beta}_o \in \mathbb{R}^L$, $o = 1, 2 \ldots, m$ weights define the neurons at the output layer. The ELM topology is show in figure 3.2. The network output is obtained with

$$\mathbf{H}\boldsymbol{\beta} = \mathbf{T} \tag{3.22}$$

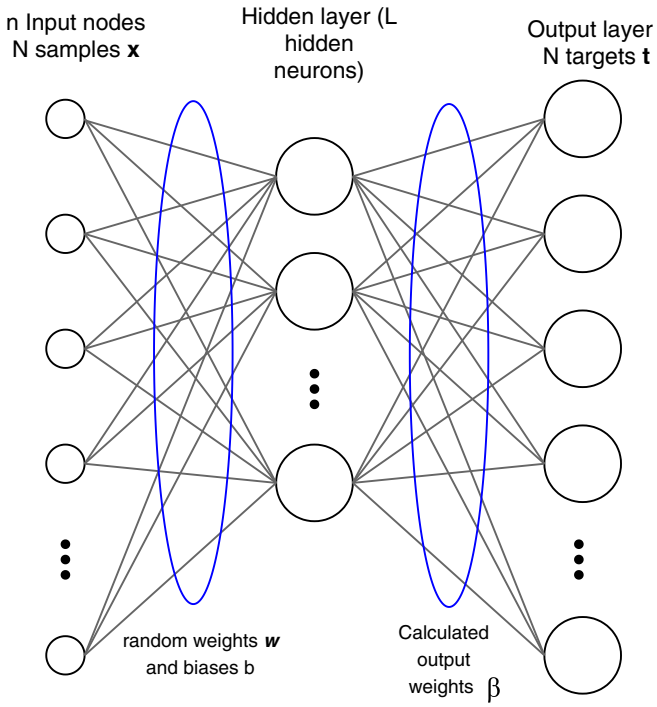


Figure 3.2: ELM general topology.

where

$$\mathbf{H}_{N\times L} = \begin{bmatrix} f(\mathbf{w_1 x_1} + b_1) & \cdots & f(\mathbf{w_L x_1} + b_L) \\ \vdots & \cdots & \vdots \\ f(\mathbf{w_1 x_N} + b_1) & \cdots & f(\mathbf{w_L x_N} + b_L) \end{bmatrix}$$

is a matrix containing the outputs of the neurons in the hidden layer, $\mathbf{T}_{N\times m}$ is a matrix with the desired outputs (targets) and $\boldsymbol{\beta}$ is the matrix containing the weights $\boldsymbol{\beta}_o$ connecting the neurons in the hidden layer to the output layer. If $N = L$, $\mathbf{H}$ is invertible [49] and the network is trained after solving equation 3.22 for $\boldsymbol{\beta}$

$$\boldsymbol{\beta} = \mathbf{H}^{-1}\mathbf{T} \tag{3.23}$$

However, if $N > L$, which is often the case, the goal for the ELM training method is [51]

$$\text{Minimize} : \| \mathbf{H}\boldsymbol{\beta} - \mathbf{T} \|^2 \text{ and } \| \boldsymbol{\beta} \| \tag{3.24}$$

which may improve the generalization performance of the network. In many cases, these are fully connected neural networks, which may render them prone to overfitting due to the large number of parameters [52]. Therefore, enhancing generalization

capability in this condition is achieved by minimizing also the norm of $\boldsymbol{\beta}$ [51]. The solution to this problem is

$$\boldsymbol{\beta} = \mathbf{H}^{\dagger}\mathbf{T} \tag{3.25}$$

where $\mathbf{H}^{\dagger}$ is the Moore-Penrose pseudoinverse of $\mathbf{H}$, which provides the smallest norm least-squares solution (therefore, smallest norm of weights $\boldsymbol{\beta}$ and minimum training error) [49]. The network is considered trained after determining $\boldsymbol{\beta}$ and all the weight connections are known. This method is usually faster than training methods based on gradient search since it relies in only one matrix equation, with no iterations nor epochs-based algorithm to adjust the weights, leaving the calculation of the Moore-Penrose inverse as the greatest burden in calculations during the training step.

The Moore-Penrose pseudoinverse can be calculated in different ways. The most used methods are [53]:

- **Singular Value Decomposition**(**SVD**). The matrix $\mathbf{H}$ can be written in the form of Singular Value Decomposition [54], which is

  $$\mathbf{H} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^{T} \tag{3.26}$$

  where $\mathbf{U}$ is an unitary matrix whose columns are eigenvectors of $\mathbf{H}\mathbf{H}^{T}$, $\boldsymbol{\Sigma}$ is a rectangular diagonal matrix whose elements are the square roots of eigenvalues of $\mathbf{H}\mathbf{H}^{T}$ or $\mathbf{H}^{T}\mathbf{H}$ and $\mathbf{V}$ is an unitary matrix whose columns are the eigenvectors of $\mathbf{H}^{T}\mathbf{H}$. The Moore-Penrose pseudoinverse is obtained by

  $$\mathbf{H}^{\dagger} = \mathbf{V}\boldsymbol{\Sigma}^{\dagger}\mathbf{U}^{T} \tag{3.27}$$

  where $\boldsymbol{\Sigma}^{\dagger}$ is the pseudoinverse of $\boldsymbol{\Sigma}$, which can be calculated by inverting the nonzero elements and then transposing the resultant matrix.

- **Orthogonal Projection**. The matrix $\mathbf{H}^{\dagger}$ can be obtained with [55]

  $$\begin{aligned} \mathbf{H}^{\dagger} &= (\mathbf{H}^{T}\mathbf{H})^{-1}\mathbf{H}^{T} \quad \text{for} \quad \mathbf{H}^{T}\mathbf{H} \quad \text{nonsingular} \\ \mathbf{H}^{\dagger} &= \mathbf{H}^{T}(\mathbf{H}\mathbf{H}^{T})^{-1} \quad \text{for} \quad \mathbf{H}\mathbf{H}^{T} \quad \text{nonsingular} \end{aligned} \tag{3.28}$$

Regarding the orthogonal projection method, adding a small positive constant to the diagonal of $\mathbf{H}^{T}\mathbf{H}$ or $\mathbf{H}\mathbf{H}^{T}$ results in stabler solution and better generalization performance [51]. The solution to this problem depends on the relationship between

the size of training set N and the hidden neurons count L [51]. In the case where $N \gg L$:

$$\boldsymbol{\beta} = \left(\frac{\mathbf{I}}{C} + \mathbf{H}^T \mathbf{H} \right)^{-1} \mathbf{H}^T \mathbf{T} \tag{3.29}$$

whereas for $N > L$

$$\boldsymbol{\beta} = \mathbf{H}^T \left(\frac{\mathbf{I}}{C} + \mathbf{H}\mathbf{H}^T \right)^{-1} \mathbf{T} \tag{3.30}$$

where $\mathbf{I}$ is the identity matrix. C is the regularization constant, which controls the network's regularization performance. It is common practice to assign large values to C (however specific for each application), so that a small constant will be added to $\mathbf{H}\mathbf{H}^T$ or $\mathbf{H}^T\mathbf{H}$.

Once finished the training step, the network can be deployed in a system, so that a presented input will be assigned to either an output value for regression problems or a class for classification problems. For multiclass ELM classifiers, which is the studied case in this work, the predicted class label for a given input sample is obtained by assigning the class corresponding to the index of the output neuron which has the highest output [51].

Although the ELM algorithm is valid for any initialization of random weights and biases in the hidden layer, it has been reported in the literature impact of different random distributions in the network performance. The work [56] experimentally verified that Uniform and Gamma distributions with small variances lead to better test performance than when using larger variances, which can compromise the overall accuracy of the network. In this regard, Chapter 6 contains the evaluation for some distributions for the studied problem in this work.

ELMs have been reported with great performance in works spread through several areas, such as forecast of soil moisture and estimation of soil organic matter (geology) [57, 58], bacteria analysis of raw goat milk (food industry) [59], classification of images according to authenticity (image forensics) [60], face recognition [61] and detection of features in astronomic images (astronomy) [62]. These works compare ELM to other classification methods (Support Vector Machines in many of them) and show comparable to superior performance of ELM.

Finally, regarding the scope of the present work, the advantages of ELM for embedded systems are twofold:

1. **Fast training**. ELM usually presents shorter training time compared to back-

propagation methods, since it doesn't require iterative weights adjustment. Usually, dedicated embedded systems run in low computational power compared with desktop computers and workstations, and intensive recursion, even with instructions such as MAC (multiply and accumulate) native to the processor, are run in slow pace due to hardware limitations.

2. **Reuse of memory**. Since the neurons at the hidden layer do not have their weights altered during the training step, their values can be stored in program memory and reused every time a new network is required to be trained. Therefore, only the values in the vector $\boldsymbol{\beta}$ need to be stored in non-volatile memory.

3.5.1 Variations in ELM

Modifications to the described ELM algorithm have been proposed since it was first proposed. Below are described the main variations [55, 63].

- Fully Complex ELM, an ELM network whose activation functions are complex.
- On-line Sequential ELM (OS-ELM), a method for training ELM networks with data samples being sequentially presented to the network without the need of batches. This network can receive training samples one by one or chunk by chunk, an the chunk size need not to be constant. Once employed in the training stage, data can be discarded prior to upcoming new data.
- Incremental ELM (I-ELM), a method for adding neurons to the hidden layer in the ELM training procedure.
- Optimally-Pruned ELM (OP-ELM), a method for eliminating hidden neurons in ELM that have low relevance to classification task.
- Kernel ELM (K-ELM), a implementation of the ELM training algorithm when the hidden layer output and number of hidden layer nodes need not to be specified to users.

3.5.2 Autoencoders

Autoencoder is a special type of ANN where the targets in the training step are the same as the inputs. The hidden layers act as an encoding step, where the inputs are transformed (encoded) into a new feature space (which may be of larger, equal or smaller dimensionality than the original input's) and then transformed back (decoded) to the input space in the output layer. This reconstruction is inherently lossy [52] in the smaller dimensionality case and the networks are trained in a manner to minimize this loss and have outputs as equal as possible to the inputs.

Autoencoders based on ELM (ELM-AE) networks were introduced in [64] and later employed in several problems [65, 66, 67]. They consist of SLFN ANNs such as the one shown in Figure 3.2 and share the training aspects of ELMs. The procedure for training ELM-AE is as follows:

1. Generate random **orthogonal** weights and bias connecting the input layer to the hidden layer. The random weights matrix $\mathbf{a}_{n\times L}$ and random bias vector $\mathbf{b}_{L\times 1}$ are such that $\mathbf{a}^T\mathbf{a} = \mathbf{I}$ and $\mathbf{b}^T\mathbf{b} = 1$. They assume, respectively, the role of $\mathbf{w}$ and b Figure 3.2.

2. Calculate the output weights $\boldsymbol{\beta}$ using either Equation 3.29 or 3.30, using the inputs for targets.

3. The new data $\mathbf{X}_{new}$ can be obtained by

$$\mathbf{X}_{new} = \mathbf{X}\boldsymbol{\beta}^T \tag{3.31}$$

Linear and nonlinear ELM-AE both follow the procedure above, and either are obtained according to the activation function of the hidden layer, whereas the decoder stage is made linear [68], not requiring tied weights (when the weights from input to hidden layer and from hidden layer to output are equal) between the layers.

3.6 Embedded systems and ANNs

3.6.1 Overview of microcontrollers

Microcontrollers (MCUs) are traditionally defined as a microprocessor unit packed with memory and interfaces on a same chip [69]. This arrangement makes such de-

vices suitable for applications that require some level of data processing or automation, such as education [70], agriculture [71] and robotics [72], with reduced hardware footprint and power consumption [73]. Microcontrollers can be found with a variety of embedded interfaces - Digital Input/Output (I/O), Timers, Analog-to-Digital Converters (ADC), Serial communication (UART, SPI, I2C) and many others.

Microcontrollers can be defined as devices designed with fixed architecture, meaning that the logic circuit blocks responsible for mathematical and logic operations can't be rearranged to new circuit blocks. In this case, the architecture is designed by the company that produces the chip. This paradigm is different for devices such as Programmable Logic Device (PLD) and Field Programmable Gate Array (FPGA), which can have their logic circuit built to match an architecture designed by the final user.

The way to use a microcontroller is to configure it through its control registers. These small memory elements dictate the behaviour of the device, following the processor instructions predefined by the user.

The task complexity that a microcontroller can carry out depends on certain aspects of its architecture:

- **Word length**. This parameter is measured in bits and it represents the general size of registers in the unit. Generally speaking, the longer the register length, more configurable is the device. The word length also establishes limits to the performance of some peripherals [74].

- **Clock**. Clock is the element that synchronizes the whole system. The synchronizing signal is fed by an stable oscillator with known frequency. The clock rating of a microcontroller is a measure of how fast the processor and peripherals can operate and dictates the performance of systems reliant on time base, such as timers and serial communication interfaces [74].

- **Throughput**. This parameter is the amount of operations a microcontroller can execute per time interval [69]. Two common measures for throughput are Millions of Instructions per Second (MIPS) or Millions of Floating Point Operations per Second (MFLOPS) in the case of devices with dedicated hardware for floating point operations. It is common to mistake throughput with

clock, since the idea of a fast clock can be followed by the idea of fast execution. While this can be true, knowing the MIPS or MFLOPS rating of microcontrollers is the proper measure of the calculation load a system can achieve.

Several models can be found in market with word lengths of 8, 16 and 32 bits, clock range from 4 to 176 MHz and many ratings of MIPS and MFLOPS [69].

3.6.2 ANNs and microcontrollers

Artificial Neural Networks trained with backpropagation error are a standard tool for modelling and classification problems in artificial intelligence. Since the commercial availability of microcontrollers, having an ANN embedded in such device was a natural outcome. However, due to the computational cost of the error backpropagation training method, the strategy for working with microcontrollers is traditionally to obtain the trained the weights in the ANN in a suitable platform, such as computer, and then port them to the microcontroller.

Many works in the literature follow this approach. Works with microcontrollers using ANNs with offline-trained weights have been reported in a variety of themes, such as classification of electrocardiogram signals for arrhythmia detection, [75], development of artificial tongue for water classification, [76], analysis of turbidity in water [77], power factor correction [78]. Except for the work in [75], which uses an ARM9 embedded platform, all others use PIC microcontrollers. In [79], a study has been carried out with a variation of extreme learning machines (kernel ELM) to identify stress conditions in virtual reality environment, but the authors state that although the setup is prepared for microcontroller port, it was not done in this work. In [80], an interface of the same type as the one used in the present work (Section 4) with kernel ELM to build an electronic nose, also with coefficients trained in MATLAB. Commercially, ARM announced a library containing kernels for common neural networks operations in fixed point [81], improving the task of porting solutions based on pre-trained deep networks.

An occurrence of training backpropagation ANNs locally in microcontrollers was found in [82], with training algorithm coded in assembly language to achieve efficiency in an 8-bit microcontroller. A network with 3 inputs, 5 neurons in hidden

layer and 1 output was used to test dynamic response of pH in a stirred tank reactor. One training iteration takes 6 s to reach completion, and the system presented good mean square error (MSE).

Although having an assembly-coded algorithm is a good measure of efficiency in calculations, assembly language is heavily dependent on the instruction set of the microcontroller since it is indeed the most lower language in the sense of closeness to machine language. This means that the solution can be poorly portable to other platforms, whereas solutions coded in C or C++ would have a broader reach.

3.7 Final Remarks

This chapter introduced classifiers based on artificial neural networks and concludes the theoretical background of this thesis. As opposed to traditional methods for training multilayer perceptron networks, Extreme Learning Machine's supposedly faster training method along with memory usage characteristics indicate its suitability for embedded applications. Methods for feature extraction were presented in order to provide samples with reduced dimensionality to the classifier, which is also desirable in embedded applications. Finally, it was presented an overview of microcontrollers and works in the literature using these devices with ANNs. Most of the works train the networks in computer environment.

In order to move the training step from computers to embedded devices, this thesis presents the feasibility of embedding extreme learning machines in microcontrollers, in order to have a decision support system for ultrasound nondestructive evaluation, capable of executing the training step in the microcontroller, instead of executing it on a personal computer environment.

Chapter 4

Proposed Methodology

4.1 Introduction

Figure 4.1 depicts a block diagram containing the architecture for the proposed decision support system using a microcontroller. In this approach, the task of data acquisition and sensor driving (exciting the ultrasonic probe at proper frequency and voltage level) are executed by the embedded system in which a microcontroller executes the training and operation tasks. This device is also responsible for digital conditioning (average removal and normalization) and execution of Fourier Transform on the data.

In the proposed approach, the storage can be done either on the microcontroller or in an external solid state integrated circuit (a popular market solution for non-volatile memory) depending on the size of the trained network. The purpose of storage is to record, in a non-volatile memory, the calculated weights for the network. Moreover, the stored content provides the network constants needed for the decision support task. This thesis assesses the blocks in Figure 4.1 concerning Dimensionality Reduction, Storage, Training Procedure and Classifier, which are described in detail in this chapter and the next chapters.

The following sections present the methodology employed in many steps of this work. The experimental setup will be detailed, as well as the types of flaws investigated in the weld bead. The main assessment tools used in this work are also presented.

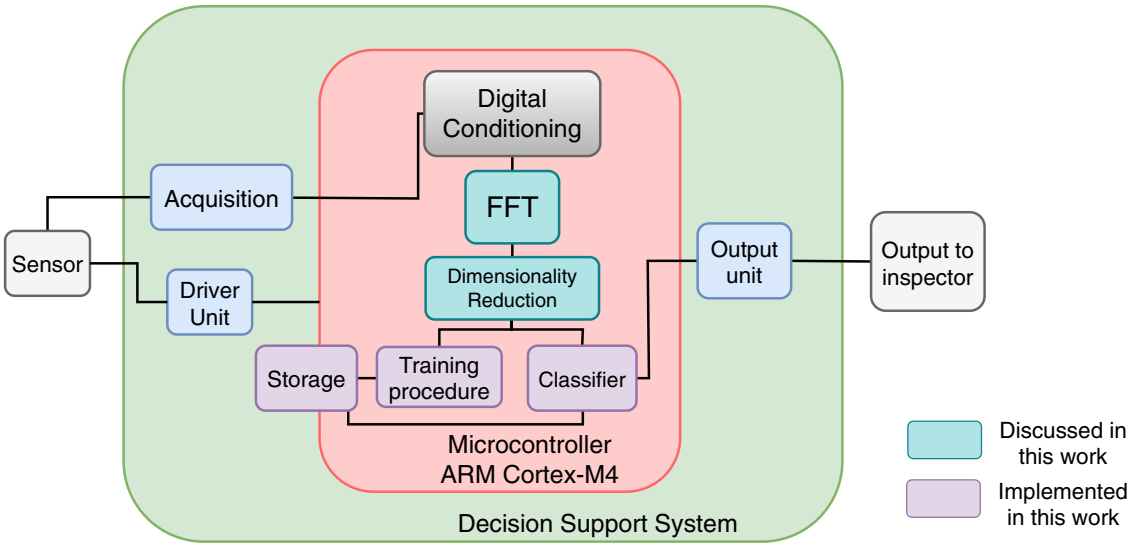


Figure 4.1: Topology for the proposed Decision Support System.

4.2 Weld defect classes

Some types of weld flaws discussed in section 2.2 are evaluated in the present case. They were divided into five categories, depicted in Figure 4.2 [1]:

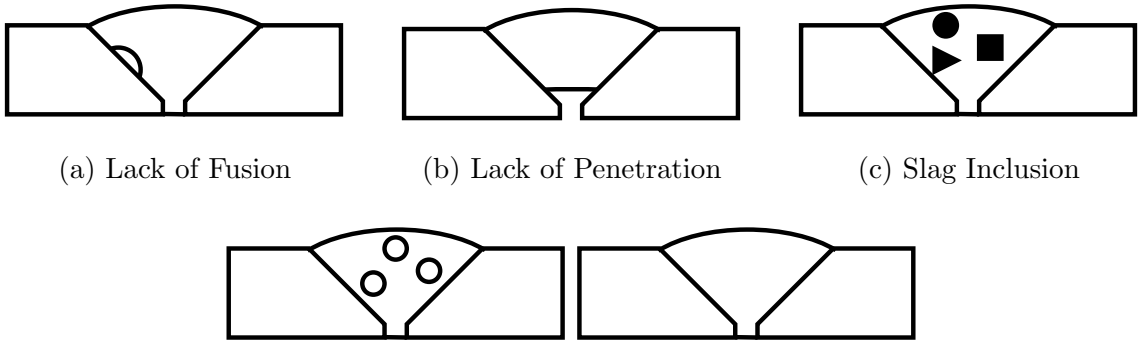
1. **Lack of Fusion (LF)**. Lack of fusion between the welding material and the base parts to be welded (see Figure 4.2a). This defect is a potential starting point for cracks within the weld.

2. **Lack of Penetration (LP)**. Lack of penetration through the entire thickness of the base metal (see Figure 4.2b). This forms a discontinuity inside the material, which can start fatigue cracking due to component use.

3. **Slag Inclusion (SI)**. Presence of solid material other than the ones involved in the welding process (weld and base metal, see Figure 4.2c). The presence of slag limits the load bearing capability of structures, therefore weakening them.

4. **Porosity (PO)**. Cavities originated from gas bubbles trapped within the welding region, which leak during solidification process (see Figure 4.2d). This is another form of discontinuity within the material that diminishes its load bearing capacity.

5. **No Defect (ND)**. Welding process done with no defects (see Figure 4.2e).



(a) Lack of Fusion (b) Lack of Penetration (c) Slag Inclusion

(d) Porosity (e) No Defect

Figure 4.2: Representation of the considered weld profiles

The defects listed above were produced in a welded junction of two carbon steel SAE 1020 plates with dimensions 1000 x 500 x 15.5 mm. The defects were inserted along the extension of the weld bead using a combination of the electrical arc welding processes GMAW, GTAW and SMAW (refer to Section 2.2). Porosity, lack of fusion and lack of penetration were introduced into the weld bead by changing the welding parameters such as current, voltage and speed. For slag inclusion, a copper wire was inserted in the weld bead during welding. Radiographic evaluation provided a map for the flaws along the weld bead, used in this work as the ground truth for classifier design.

4.3 Experiment Setup

Figures 4.3 to 4.5 present the experiment setup. Figure 4.3 depicts a schematic of the experiment set up in order to acquire samples from the specimen and build the data set. The pulse generator acts as excitation source for the ToFD probes positioned at the specimen. The ToFD signal readings from the oscilloscope are recorded in the computer. Figure 4.4 shows the actual specimen and how the defects are mapped along the weld bead. Figure 4.5 shows the probes positioning in the ToFD experiment being executed at the specimen.

The experiment was carried out using 3 mm crystal transducers with acrylic wedges operating at 5 MHz and wavelength corresponding to 648 μm . The source signal was a square-shaped pulse with seven wave periods per pulse, at 100 V peak

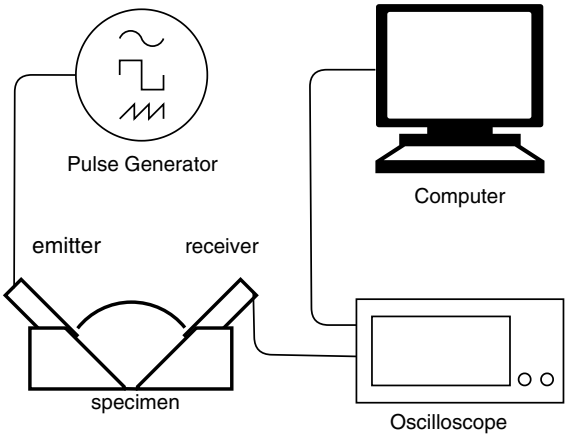


Figure 4.3: Experimental setup.

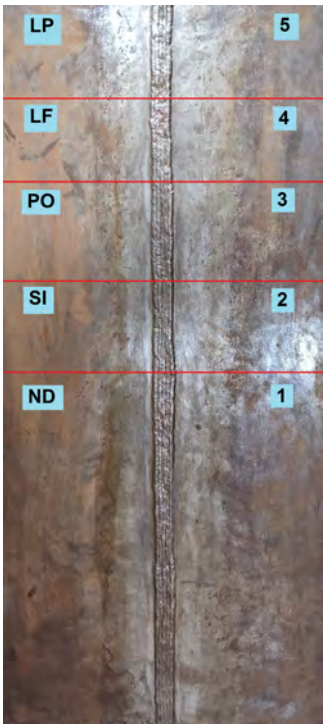


Figure 4.4: Mapping of flaws inserted along the specimen.

Figure 4.5: Specimen and ToFD ultrasonic probes.

voltage (10 dB gain). The pair emitter-receiver was displaced alongside the region of interest (the weld bead, which is 36 mm wide), positioned with crystal centers at a 62 mm skip distance. The incidence angle θ was 63 degrees. Each signal was collected at an oscilloscope and recorded for further processing in a personal computer. Each ToFD signal consisted of 2500 time samples. These measures are

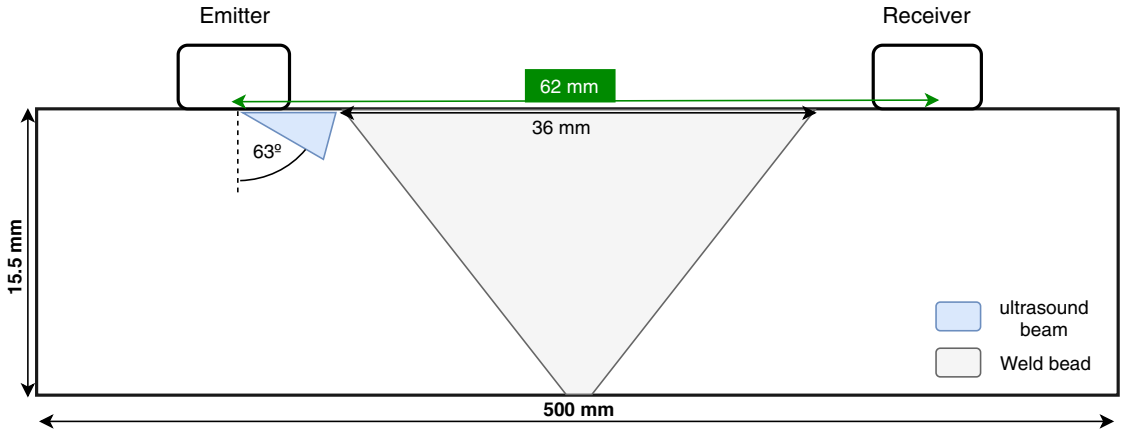


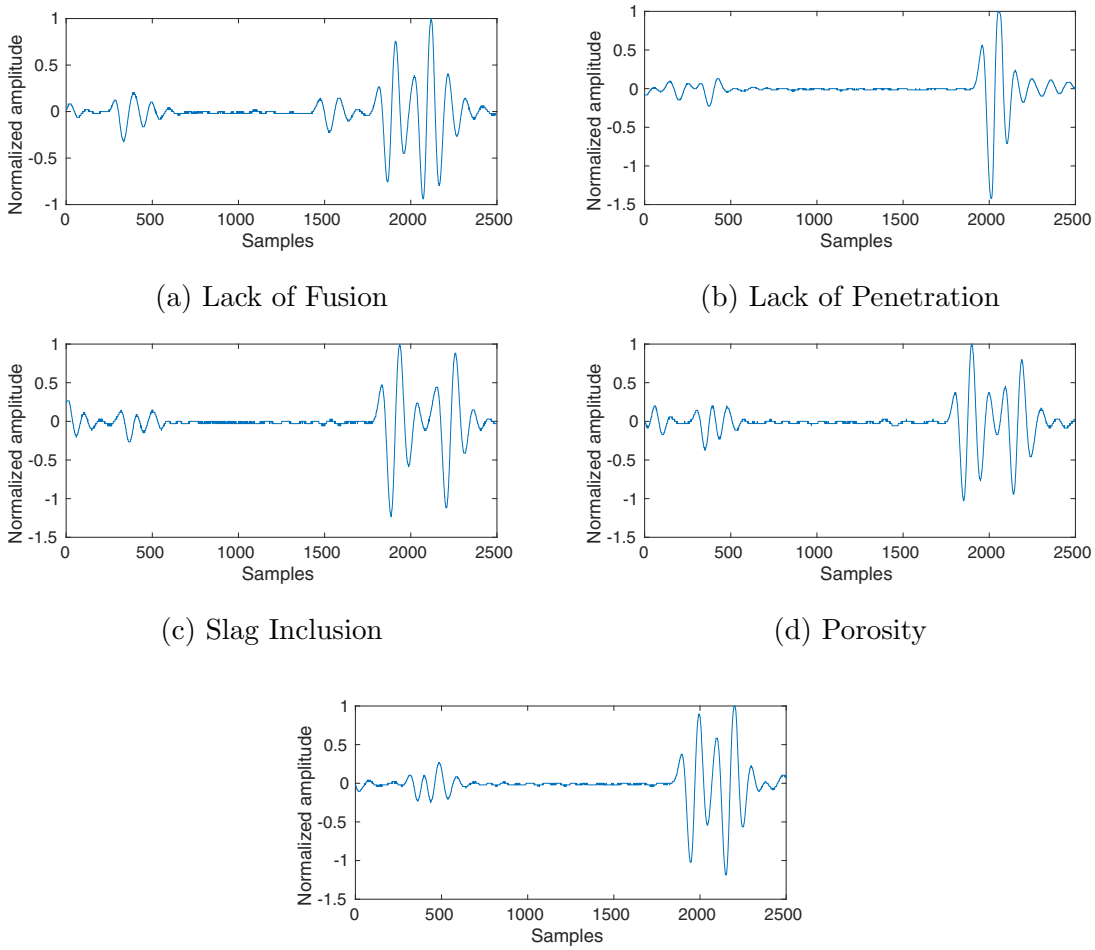
Figure 4.6: Setup measures in ToFD experiment (out of scale).

depicted in Figure 4.6.

The signals were sampled at 500 MHz. This sampling rate, the wave length and the layout in Figure 4.6 result in the following lengths for each segment: 700 time samples for lateral wave, 750 time samples for backwall waves and 1050 samples for diffraction waves interval. These intervals comprise the duration of lateral and backwall pulses and the time in between for diffraction wave occurrence.

Figures 4.7a to 4.7e show typical examples of ToFD signals acquired at the specimen using the described setup. The time-domain waveforms show similarities between each other with slightly different features that can be noticed by looking at the lateral and backwall waves (refer to Figure 2.6). The classes Slag Inclusion (Figure 4.7c) and Porosity (Figure 4.7d), due to the similarity in its physical nature, share similiar typical waveforms, which present two peaks surrounding a phase inversion in the backwall waves. The classes Lack of Fusion (Figure 4.7a) and No Defect (Figure 4.7e) both exhibit two peaks in the backwall wave, with minor differences in the lateral wave. The class Lack of Penetration (Figure 4.7b) presents shorter wave duration than the other classes, possibly due to attenuation in the air interface between the specimen and the weld bead.

A total amount of 1000 signals were acquired at the carbon steel laminate, which were equally split among all classes, resulting in 200 signals for each class.



(a) Lack of Fusion

(b) Lack of Penetration

(c) Slag Inclusion

(d) Porosity

(e) No Defect

Figure 4.7: Example of typical signals acquired at the specimen.

4.4 Signal processing chain

The evaluation of the decision support system in this thesis consists of applying the signal processing chain in Figure 4.8 to the ToFD ultrasound samples. The block "segmentation" in Figure 4.8 represents the segmentation of ToFD signals, which is a new approach proposed in this work, further discussed in Section 4.4.1.

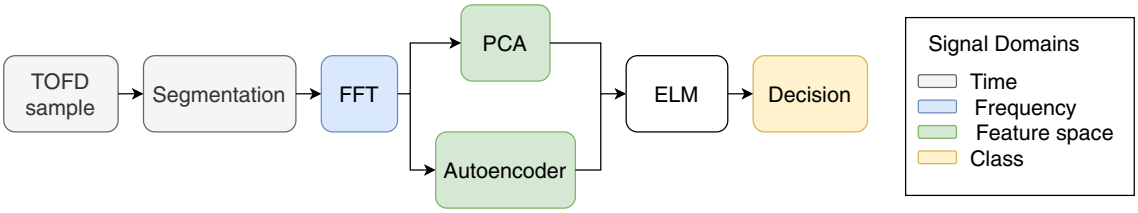


Figure 4.8: Signal path in the decision support system.

Although Figure 4.8 shows the whole processing chain designed for a sample classification, in this work some blocks were evaluated separately in the following order:

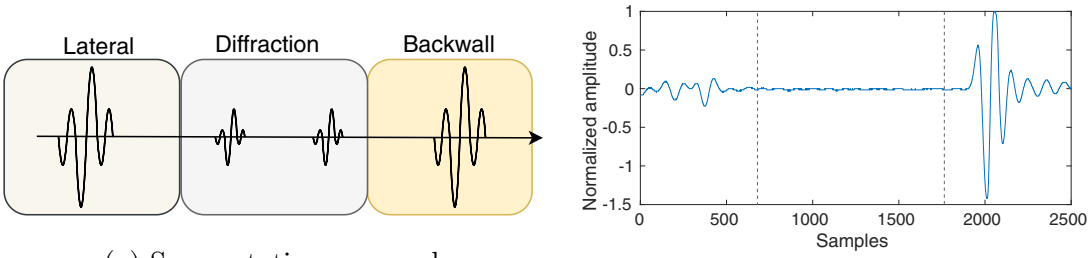
1. ToFD sample → FFT → ELM, for evaluation of ELM as classifier solution;
2. ToFD sample → Segmentation → FFT → ELM, for evaluation of the segmentation approach;
3. The whole chain in Figure 4.8, for evaluation of PCA and Autoencoder as feature extraction techniques;
4. The whole chain in Figure 4.8, for evaluation of parameters of the ELM classifier for the purpose of embedding it in microcontrollers.

Items 1 and 2 are detailed in Chapter 5 and items 3 and 4 are detailed in Chapter 6.

4.4.1 Segmentation approach

The segmentation of ToFD samples is proposed in order to evaluate the effect of the distinct ToFD wave features described in Section 2.5 on defect classification. Each ultrasound wave sample was segmented into the three categories described in Section 2.5, namely Lateral Wave, Diffraction Waves and Backwall Wave. Figure 4.9 shows the proposed approach, where Figure 4.9a sketches the proposed approach for a generic ToFD signal and Figure 4.9b shows the segmentation of a typical LP signal of the data set. Since the goal is to deal with frequency magnitude spectrum, the segmentation must be executed considering the envelope of the waves (which tend to 0 value at both ends), in order to avoid problems with spectral leakage [35]. Since lateral and backwall waves already present windowed behaviour, no windowing method was employed prior to calculating the magnitude spectrum.

The reason behind the idea of segmentation, is the reduction of feature set at the input of the neural classifier. It was expected in this work that the diffraction waves in ToFD technique would result in little to no influence in the task of assigning classes to ToFD inputs.



(a) Segmentation approach.

(b) Segmentation in a typical LP signal.

Figure 4.9: Segmentation approach of ToFD signal.

Diffraction waves are useful in flaw sizing, since their appearance in the measurement time frame provide information about the flaw tips, because the phenomenon of diffraction is inherently linked to waves propagating after reaching tips. However, the types of flaws investigated in this work do not follow the pattern of a single crack or flaw crossing the weld bead. Some cases, such as slag inclusion and porosity, there is too much dispersion of incident discontinuities through the cross section of the weld bead. On the other hand, the lateral wave and backwall wave cover a great area in the weld bead cross section, so that the respective received signals carry relevant information about the flaws described in Section 4.2.

Concerning an embedded decision support system, the strategy of segmentation can be achieved with prior information of characteristics of the specimen, such as plate thickness, material type and probe displacement. This quantities can be used in the calculation of time intervals needed to trigger the sampling of lateral, diffraction and backwall waves, accordingly, upon exciting the sensor:

- Lateral waves will reach the receiver after

$$t_l = \frac{\text{skip distance}}{\lambda f}$$

  where t_l is the time in seconds, skip distance is the distance between transmitter and receiver probes, λ is the ultrasound wavelength and f is its frequency.

- Backwall waves will reach the receiver after

$$t_b = \frac{2 \cdot \text{thickness}}{cos(\theta)\lambda f}$$

  where t_b is the time in seconds, thickness is the specimen's thickness, θ is the incidence angle between the ultrasonic beam and the specimen surface's normal vector, λ is the ultrasound wavelength and f is its frequency.

- Diffraction waves may occur at any time t in the interval $t_l < t < t_b$

4.5 Evaluation and Testing

Since this work assesses features in the design of a decision support system for the aforementioned weld defects, classifiers based on artificial neural network will be evaluated throughout the next chapters. The goal is to leverage ELMs in microcontrolled systems using the advantages mentioned in Section 3.5. ELM will be compared with traditional backpropagation-based training methods for ANNs, namely Levenberg-Marquardt and RPROP considered fast implementations of the backpropagation algorithm (see Section 3.4.1).

The following metrics will be used in order to evaluate classifiers:

- **Test accuracy**. The test accuracy TA will be represented as the ratio of correctly assigned test signals N_C to the total amount of signals N_T in the test set. The average test accuracy and maximum test accuracy will be evaluated.

$$TA = \frac{N_C}{N_T} \tag{4.1}$$

  Accuracy is an important metric to understand the performance of a decision support system, because such system must aid an inspector in the decision regarding the specimen's integrity. Moreover, accuracy is an important tool to assess the system's rate of false negatives, since an overseen defect in an specimen's structure may put assets and human lives at risk.

- **Training time**. This is the time interval taken to complete the training procedure of the classifier. Since this characteristic depends on the computational resources available where the training procedure occurs, this quantity will be evaluated only by comparing the training times among the methods executed in a same computational environment, disregarding the absolute quantity as a measure.

- **Confusion matrix**. The confusion matrix for the best accuracies obtained from each classifier will be evaluated (refer to Section 3.2).

The methodology used to train the classifiers proceeded according to the following steps (see also Figure 4.10):

1. Divide the signals into 10 randomly sorted sets, each set with the same number of signals for each class.

2. Categorize the sets into training, validation and test sets in the proportion 50%, 30% and 20% of the total amount of samples respectively for backpropagation-based methods. For the ELM case, since the method doesn't have validation step, only training and test sets were used, in the proportion 70% and 30% of the total amount of samples respectively.

3. Initialize 50 instances of classifiers and do the training and test steps with the current arrangement of training set and test set for each instance. Evaluate the accuracy and confusion matrix for each instance of classifier.

4. Redefine the training set and the test set among the 10 sets of samples and go to step 2, until 10 redefinitions of the training and test sets take place. The process ends with 500 evaluations of accuracy and confusion matrix for the classifiers.

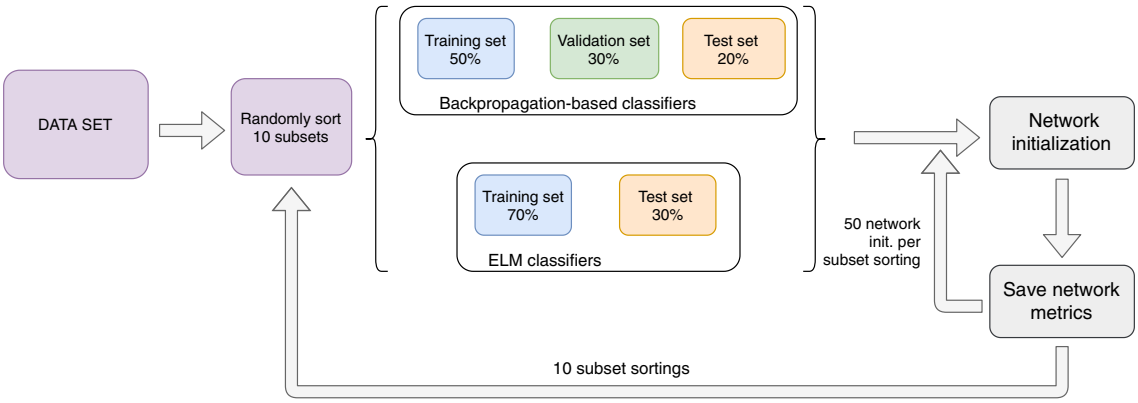


Figure 4.10: Methodology for classifier evaluation.

This method tries to avoid biased performances due to specific arrangements throughout sets of samples.

The evaluation of classifiers as well as their parameters were carried out in MATLAB environment in a linux 64-bit running in a personal computer whose specification is Intel® Core™ i5-3337U CPU @ 1.80 GHz × 4, 6 Gb RAM. The data set

used is the same described previously in Section 4.3. Dimensionality reduction in the data set was also executed in virtual environment. Once the best parameters for classifier implementation in the microcontroller were determined, they were ported to the device by means of a designed application in C++. Experiments concerning the embedded classifier in the microcontroller were executed using the same inputs as the ones for classifier evaluation in MATLAB. These inputs were made available to the microcontroller by means of serial data transmission from the personal computer to the embedded platform.

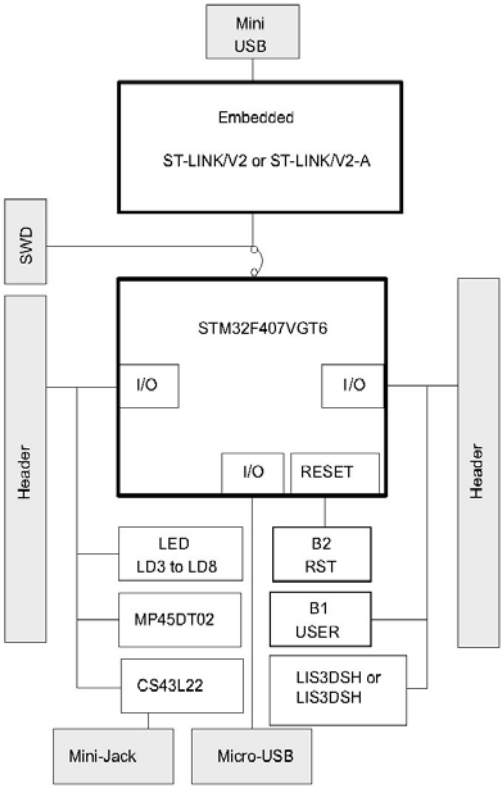
The chosen microcontrolled platform for this work is the family STM32F4 from ST Microelectronics. STM32s are abundant in market for 32-bit microcontrollers (examples of commercial applications are GPS transceivers [83], motor controllers [84] and patient monitoring devices [85]), and offer several models with different packages and configurations, all of them with ARM cores at their CPUs. The family STM32F4 has arithmetic and logic units (ALUs) with instructions for floating point operations executed in dedicated hardware. Evaluation boards are also available to shorten development time.

The actual microcontroller model is STM32F407 and ST Microelectronics provides it with a platform called STM32F4 Discovery (Figure 4.11), which has the microcontroller, an in-circuit flash programmer and other accessories such as on board accelerometer and user LEDs (Figure 4.11a). This was the platform used in the tests, since it comprises, in a single printed circuit board, the microcontroller, an embedded programmer and serial communication transceiver (Figure 4.11b), which enables the communication between the device and a personal computer.

4.6 Final Remarks

This Chapter provided detailed information about the acquisition of the data set used to train and test the embedded classifier, the methods for building the decision support system and metrics for performance evaluation. The characteristics of the considered weld defects and the arrangement for measuring them were shown in detail, and the typical signals for each class indicate that the diffraction segment may be of low value for classification, being the reason of a proposed new segmentation

(a) Microcontrolled platform. From https://www.robotistan.com/stm32f4-discovery-eng



(b) STM32F4 Discovery - block diagram. From https://microcontrollerslab.com/stm32f4-discovery-board-pinout-features-examples/

Figure 4.11: Overview of STM32F4 Discovery.

approach for handling ToFD signals.

The proposed architecture for an embedded decision support system was represented, as well as the entire cycle of signal processing from the ToFD sample to the classification task. Finally, the chosen microcontroller was presented.

The evaluation and respective results mentioned in this Chapter are the subject of the next two Chapters. The performance of ELM for ultrasound NDE and the performance of ToFD segmentation are evaluated in Chapter 5, whereas Chapter 6 has the discussion about the feature extraction methods and the aspects of implementation of ELM classifier in microcontroller.

Chapter 5

Classification performance and ToFD analysis

5.1 Introduction

In this Chapter, it is presented a comparison between the ANNs trained with the backpropagation algorithm and ELM neuron networks, in order to assess the suitability of the latter for the classification of weld defects, since most works shown in Section 2.7 use ANN classifiers with backpropagation training. The backpropagation-trained networks follow the modifications to the algorithm mentioned in Section 3.4.1 (Levenberg-Marquardt and RPROP).

Moreover, in a further discussion about the frequency content analysis of ToFD signals and the way it is used in the literature, it is proposed a new segmentation method for analysing this kind of signal, which features one of the contributions of this thesis.

5.2 Suitability of ELM for Ultrasound NDE decision support

As already mentioned, many works in the literature use ANN classifiers trained with the backpropagation algorithm. Their characteristic training relying on iterative gradient optimization of the neuron weights demands great computational effort as shown in section 3.4. ELMs, on the other hand, have a rather faster training step by

relying on analytic matricial solution, but no work has been found in the literature employing it in the NDE context in literature.

The method for sorting sample sets detailed in Section 4.5 was used to provide inputs to the selected training algorithms for ANNS, LM, RPROP and ELM.

The networks were trained using a MATLAB toolbox for neural networks. LM and RPROP methods were executed using the standard training function provided by the environment. The parameters used in both functions are summarized in Table 5.1, where common parameters were adjusted to the same values in order to evaluate the methods under similar performance constraints. A toolbox for ELM training provided by the authors of [49] was used in the ELM case.

Parameter	LM	RPROP
Max. epochs	1000	1000
Performance goal	0	0
Max. validation failures	6	6
Minimum performance gradient	$1e-5$	$1e-5$
Initial μ	0.001	—
μ decrease factor	0.1	—
μ increase factor	10	—
Max. μ	$1e10$	—
Max. time to train in seconds	inf	inf
Learning rate	—	0.01
Initial weight change	—	0.07
Increment to weight change	—	1.2
Decrement to weight change	—	0.5
Max. weight change	—	50

Table 5.1: Parameters used in LM and RPROP trainings in MATLAB

5.2.1 Feature selection

The features chosen to serve as inputs to the classifiers are the discrete frequency magnitude spectrum components (obtained from the Discrete Fourier Transform).

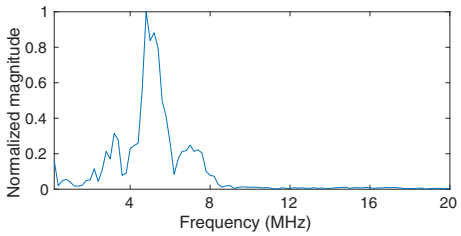
The spectrum of NDE signals has been used as feature to classifiers as shown in some works in Section 2.7. The advantage of this approach is that instead of using the entire time signal to estimate relevant features, they can be extracted directly from the frequency components of interest, resulting in a more compact feature set. Also, because the source signals are real-valued, half magnitude spectrum is redundant, and the regions of high frequency, which have almost no energy content in this case, may have irrelevant contribution. Moreover, in [86] it is shown that DFT presents the best performance in classification systems for weld defects when compared to other digital signal processing techniques such as Discrete Cosine Transform (DCT) and Wavelets (the latter though has been reported to be useful in noisy environments [87]).

Another advantage of using frequency content is the increasing availability of microcontrollers that run DFT (in the form of a faster algorithm called Fast Fourier Transform) for frequency analysis with dedicated peripherals (hardware) for this kind of task. This feature favors the faster computation of the complex products in the Discrete Fourier Transform's sum, meaning that obtaining a feature set based on frequency content is less resource-demanding than other methods, as they should require an entire software-based computation method for a solution to be executed in the microcontroller.

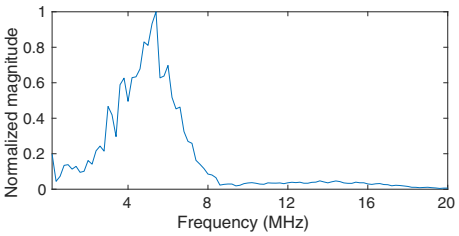
Figures 5.1a to 5.1e show the typical frequency magnitude spectra of ToFD signals acquired at the specimen corresponding to the positive frequency magnitude spectra computed through DFT for the time-domain signatures shown in Figures 4.7a to 4.7e.

5.2.2 Classifier Evaluation

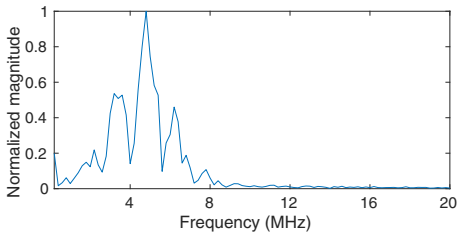
Figure 5.2 depicts boxplot graphics showing best accuracy performances obtained in each of the ten set sortings in the evaluation methodology described in Section 4.5. The graphics correspond to the ELM network and the networks trained with Levenberg-Marquardt and RPROP methods, using the spectrum of the ToFD signal. The ELM network was initialized with random weights generated by uniform distribution in the interval [-1, 1]. For this comparison, the best performances were considered in order to compare how well ELM could perform when compared to



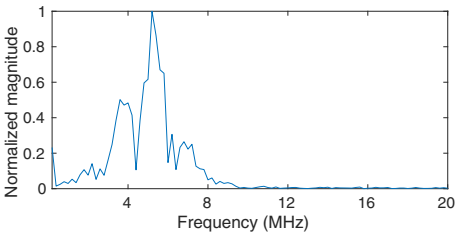
(a) Lack of Fusion



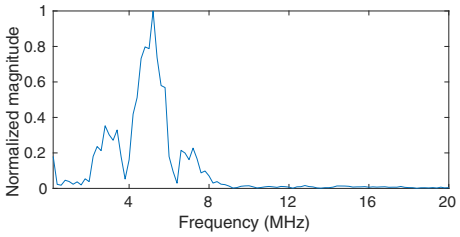
(b) Lack of Penetration



(c) Slag Inclusion



(d) Porosity



(e) No Defect

Figure 5.1: Example of positive frequency spectra obtained from typical samples acquired at the specimen.

the other methods. The boxplots show that the LM algorithm produces the best accuracy performances. The RPROP algorithm is the least disperse of all methods, however at a lower median accuracy than LM. ELM algorithm produced maximum accuracy as high as LM, but its best accuracy performances show more dispersion.

Tables 5.2a to 5.2c show the average confusion matrices obtained from the best accuracy performances shown in Figure 5.2, for each algorithm. In Table 5.3, there is a comparison between the training methods in terms of accuracy, relative training time and number of neurons in the hidden layer.

As seen in Tables 5.2a to 5.2c and Table 5.3, the results indicate that the ELM accuracy is similar to RPROP and a few percentage points close to LM for certain classes. LM provided the best accuracies for LP, PO and ND. RPROP presented poorer accuracy performance for LF, LP and SI, and is more close to ELM in ND

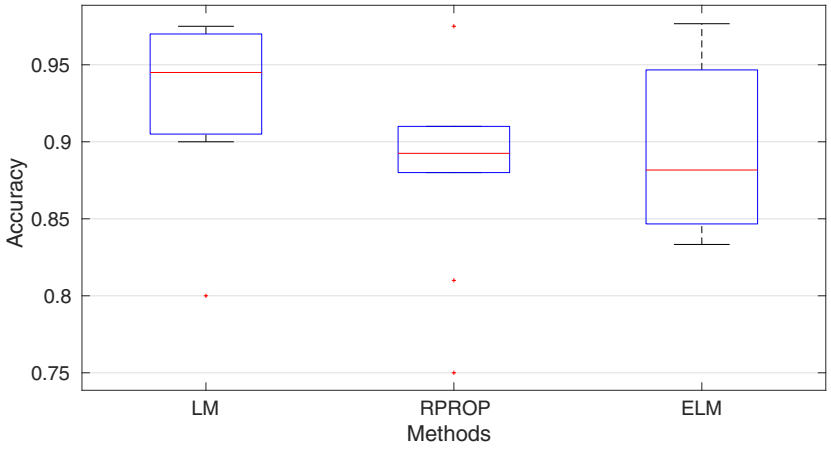


Figure 5.2: Boxplots of the maximum accuracy values obtained from tests in the data set for each training algorithm.

accuracy than it is to LM. ELM provided the best accuracy performance for LF class. All training methods reach similar maximum accuracy performance, around 97%.

Table 5.2: Average best confusion matrices for the considered training methods.

(a) LM

%	LF	LP	SI	PO	ND
LF	**86.3**	9.3	1.7	1.7	1.0
LP	1.0	**96.5**	0.0	0.0	2.5
SI	0.0	0.0	**89.3**	6.5	4.2
PO	0.2	0	3.8	**96.0**	0
ND	0.0	1.5	1.2	0	**97.3**

(b) RPROP

%	LF	LP	SI	PO	ND
LF	**82.5**	10.0	3.7	2.5	1.3
LP	9.0	**88.0**	1.3	0	1.7
SI	0.0	0.0	**82.0**	15.5	2.5
PO	0.0	1.3	5.2	**93.5**	0.0
ND	0.5	3.0	2.5	0.0	**94.0**

(c) ELM

%	LF	LP	SI	PO	ND
LF	**89.2**	2.8	0.0	1.7	6.3
LP	5.0	**88.2**	0.0	1.0	5.8
SI	3.0	2.5	**85.8**	8.7	0.0
PO	0.8	0.1	7.8	**90.7**	0.0
ND	1.5	3.5	2.7	0	**92.3**

In the problem of NDE of structures, it is relevant for classifiers to have low false negative rate, which in the current problem translates into less defective samples

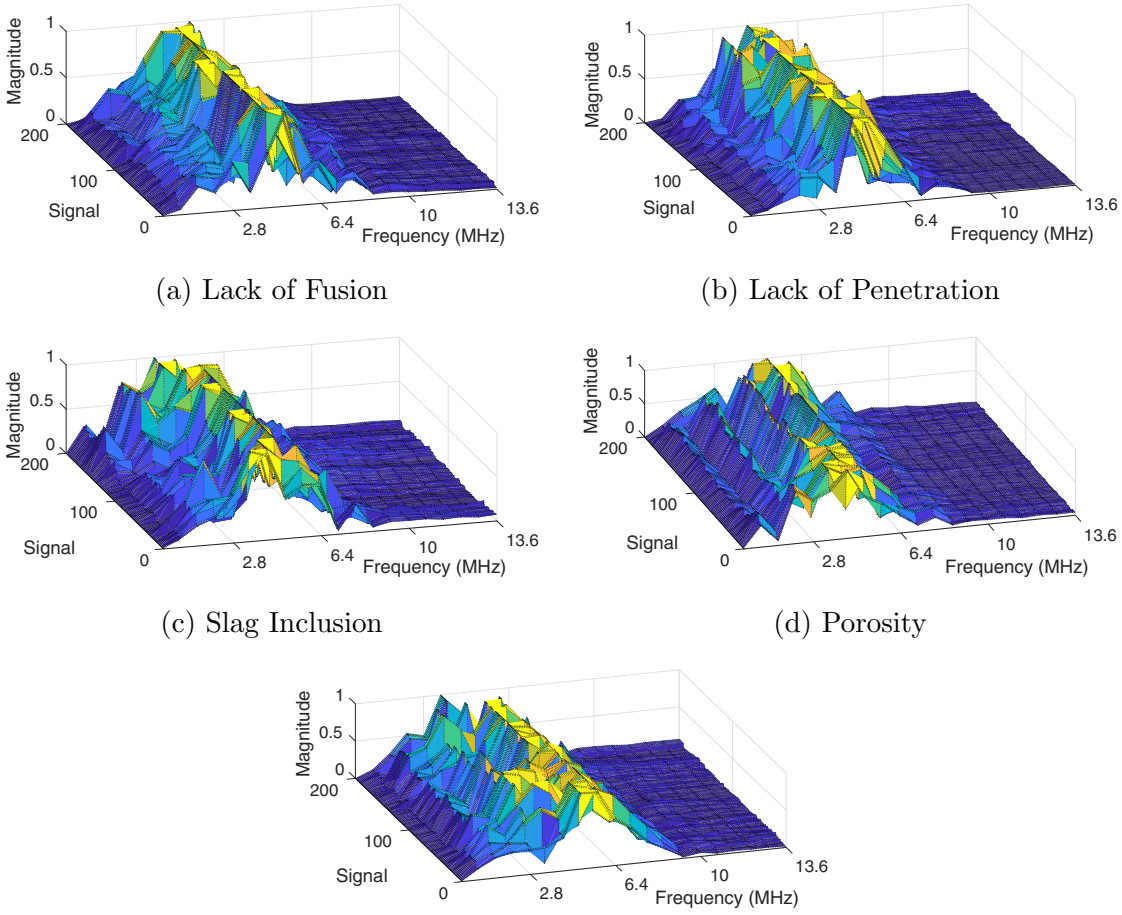
Table 5.3: Comparison between training methods.

Training method	**Average best accuracy**	**Best accuracy**	**Training time based on ELM**	**Hidden layer neurons**
LM	93.1%	97.6%	1250	20
RPROP	88.9%	97.5%	25	20
ELM	89.2%	97.5%	1	95

being assigned as flawless samples. All classifiers presented low misclassification percentage regarding the ND class. Although no algorithm has assigned PO as ND, ELM algorithm was the only to do the same for SI, at the expense of higher percentage points in LF and LP when compared to the other methods.

However, when comparing the relative training times, the ELM network stands out. The training time for ELM is smaller by a factor of 10^3 when compared to Levenberg-Marquardt method, and 25 times faster than the RPROP method. Therefore, the results show that ELM is comparable in accuracy and is superior in training speed to the LM and RPROP methods when using the frequency content of the ToFD signal regarding the present flaws. This fact indicates that ELM-trained classifiers for the current problem may have better timing performance than backpropagation methods in resource-limited computational environments such as embedded systems, which is considered in this thesis an acceptable and promising trade-off for the slight less accuracy performance when compared to other networks.

Nevertheless, as seen in Table 5.3, in order to match the discrimination performance of backpropagation-based methods, the ELM training requires a greater commitment to the neuron count, resulting in more neurons in the hidden layer. In the current comparison, after iteratively increasing the neuron count at the hidden layer, the 95-neuron hidden layer network was considered as of comparable accuracy performance as the other considered backpropagation methods. This will also imply in elapsed time during operation, because, although execution is independent of training method, the number of operations is proportional to the neuron count, which tend to be higher in ELM-trained networks, as indicated by Table 5.3. However, the methods in Section 5.3 and in Chapter 6 provide strategies for lowering



(a) Lack of Fusion

(b) Lack of Penetration

(c) Slag Inclusion

(d) Porosity

(e) No defect

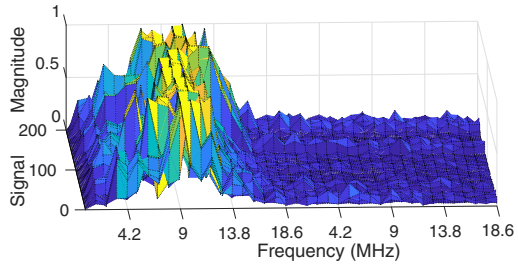
Figure 5.3: Magnitude spectra for lateral waves

the neuron count in ELMs for the problem studied in this work.

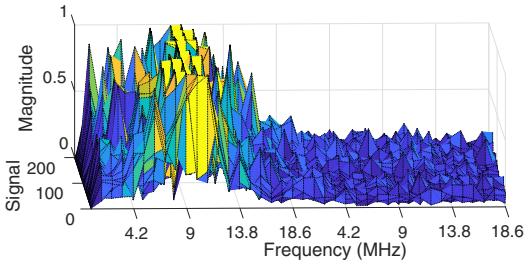
5.3 ToFD Segmentation Analysis

Given the suitability of ELM classifiers shown in section 5.2.2, they were re-trained using each category of segmentation and the results were compared to the network trained with frequency content from the entire ToFD signal.

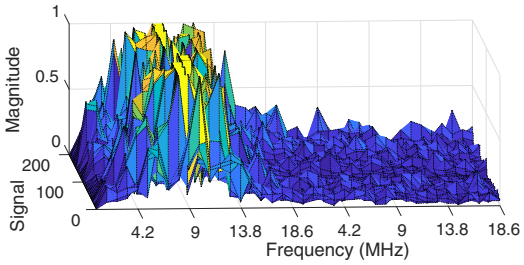
Figures 5.3, 5.4 and 5.5 show the magnitude frequency content of the segmented signals for each class of interest in terms of frequency bins, which are the indexes of the DFT outputs. Figure 5.3 contains the surfaces formed by the magnitude spectral content for the lateral wave segment in each ToFD sample in the data set, and their aspect for each class of interest. In a similar way, the same applies for diffraction waves in Figure 5.4 and backwall waves in Figure 5.5.



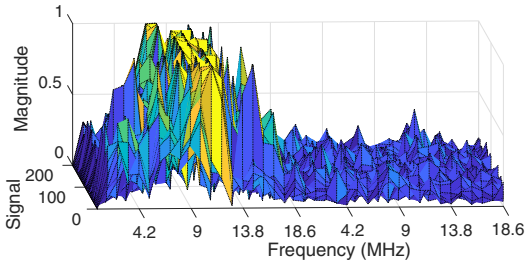
(a) Lack of Fusion



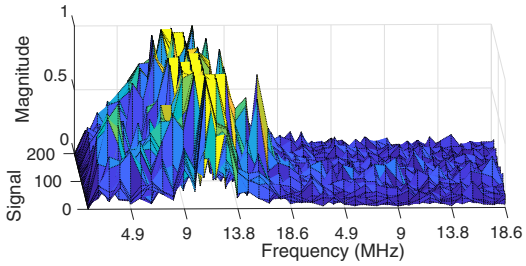
(b) Lack of Penetration



(c) Slag Inclusion



(d) Porosity

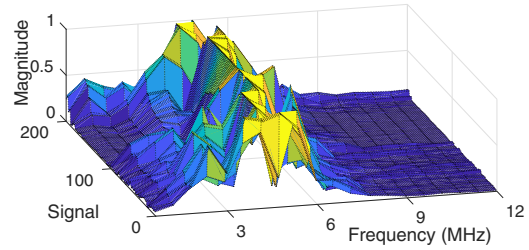


(e) No defect

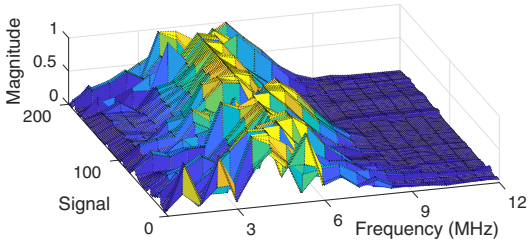
Figure 5.4: Spectra for diffraction waves

It is possible to see in both lateral wave and backwall wave spectral surfaces that within the available sample sets, some class-discriminant patterns can be determined. These discriminant patterns are the main spectral lobe around the probe frequency (5 MHz), whose width vary among the classes' spectra, and the side lobes present around the main lobe, which vary in amplitude and quantity throughout the classes. Some examples are listed in the following:

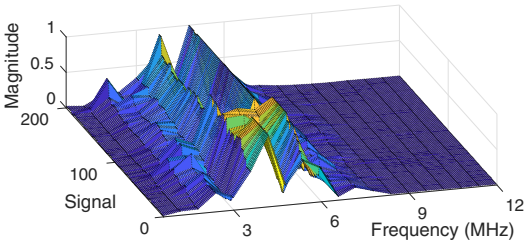
- The lateral spectra surface for Lack of Penetration (Figure 5.3a) presents narrow lobe around peak frequency when compared to other surfaces;
- Porosity surface show characteristic side lobes throughout lateral (Figure 5.3d) and backwall (Figure 5.5d) magnitude spectra;
- Lateral and backwall surfaces in No Defect class (Figures 5.3e and 5.5e) exhibit a large main lobe;



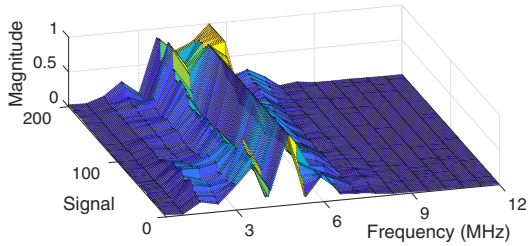
(a) Lack of Fusion



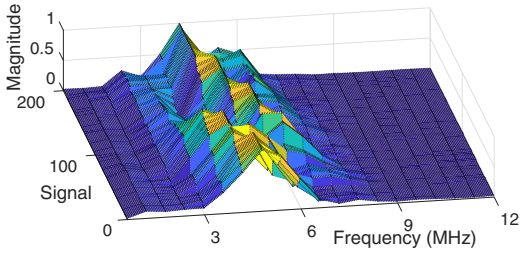
(b) Lack of Penetration



(c) Slag Inclusion



(d) Porosity



(e) No defect

Figure 5.5: Spectra for backwall waves

- Slag Inclusion and Porosity classes are the ones with the most constant spectral envelope inside the data set for lateral and backwall waves, comprising distinct side lobes.

In the case of diffraction waves, it was not possible to establish differences between the classes of interest based only on spectral information. The spectral surface for all classes are quite similar. Also, comparing with the other surfaces for lateral and backwall waves, the spectral content is remarkably different, which is an indication that the ToFD signal is non stationary. The implications of this phenomenon are:

1. The diffraction waves in the evaluated defects don't have enough energy to match the levels of the lateral and backwall waves. Their presence, if any, excites the sensor at very low levels, entering in the quantization error zone as

seen in Figures 4.7a to 4.7e.

2. Using the Fourier Transform of the entire ToFD signal in these conditions is a inaccurate procedure, since the signal changes its dynamics over the interval of the diffraction waves. Thus, the segmented analysis proposed in this work seems to provide a more appropriated characterization of the ToFD information.

3. The diffraction waves may offer little to no contribution to class separation for the considered specimen, under the spectral content point of view.

The accuracy obtained from ELM classifiers for different input features is present in Figure 5.6. The ELM network was trained with 60 neurons in the hidden layer (a number chosen after exhaustive testing by incrementing the hidden layer's neuron count) and 5 neurons in the output layer (one for each class). For comparison, the results for the ELM network trained using the frequency spectrum of the whole signal was included (the same as the one shown in Figure 5.2, here containing all test results, as opposed to the best accuracies in the original figure). Table 5.4 has the input node count for the classifier trained with ToFD magnitude spectra and each classifier trained with segmented spectra.

Table 5.4: Input nodes for the segmented classifiers.

Parameter	ToFD	Lateral	Backwall	Diffraction	Backwall + Lateral
Input nodes	90	20	20	20	40

It can be seen in Figure 5.6 that training the network only from lateral or backwall segments doesn't improve the overall accuracy performance. Also, training a network exclusively from diffraction signals produces poor classification efficiency, as expected for the nature of these signals in the current problem. However, removing the diffraction waves segment from the signal used as input leads to a reduction in dispersion in accuracy performance, comparing the boxplots corresponding to ToFD and lateral+backwall, keeping the latter at about the same median level as the former.

The combination of lateral and backwall spectra presented concentration around the median obtained for ToFD signal. The average value for accuracy in the Lat-

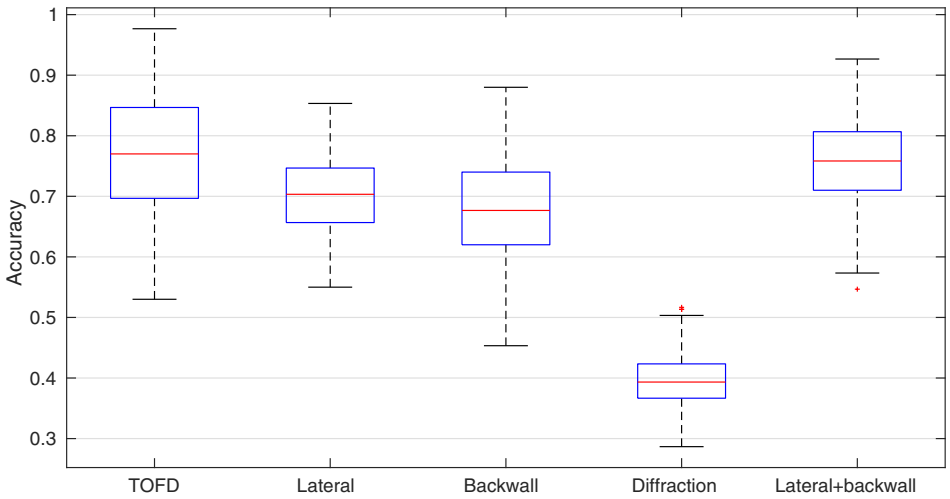


Figure 5.6: Accuracy boxplots of the segmentation-trained networks and the entire TOFD signal in Figure 5.2

eral+backwall case is 75.7%, which is similar to the values reported in [2] and [27] (both around 75%). The maximum accuracy obtained in the present work using segmented signals (92.67%) is higher than the one reported in [2] (86%), although, compared to the latter, the present work used only frequency spectrum as input to the network, resulting in a smaller dimensionality at the classifier's input layer and also featuring no other input preprocessing besides the DFT. The higher concentration of the accuracy performance though trades for a slight lower maximum accuracy performance than the one seen using ToFD signal.

In Table 5.5a is the average confusion matrix for the system trained with the spectrum of the whole ToFD signal, and the average confusion matrices obtained in the network tests for different input features are shown in Tables 5.5b to 5.5e. The average performance in classification for networks trained only from Lateral and from Backwall waves is poorer for some classes (Slag Inclusion and No defect, respectively). However, the network trained from the combination of both signals, shown in Table 5.5e follows the trend seen in Tables 5.2a to 5.2c, which is the highest average accuracy performance in Porosity and No Defect class.

In order to show the resulting dimensionality reduction from the segmentation approach, in Figure 5.7 it is depicted, as an example, the frequency spectra for a typical Lack of Penetration signal, showing magnitude spectra for lateral wave,

Table 5.5: Average confusion matrices.

(a) TOFD.

%	LF	LP	SI	PO	ND
LF	**69.0**	10.1	6.7	4.0	10.1
LP	13.0	**73.8**	3.3	2.5	7.2
SI	1.9	1.9	**76.2**	15.8	4.1
PO	4.1	2.7	3.7	**88.9**	0.5
ND	6.4	9.1	4.6	1.3	**78.6**

(b) Lateral wave.

%	LF	LP	SI	PO	ND
LF	**63.5**	11.9	14.1	3.3	7.0
LP	19.8	**67.0**	5.2	1.0	6.9
SI	11.6	10.7	**58.4**	11.8	7.5
PO	4.8	0.6	9.3	**83.8**	1.3
ND	7.3	8.1	4.6	3.0	**77.1**

(c) Backwall wave.

%	LF	LP	SI	PO	ND
LF	**39.0**	15.6	9.7	9.0	26.6
LP	12.1	**67.9**	5.3	4.5	10.3
SI	1.2	0.4	**83.1**	14.3	0.6
PO	2.1	0.0	9.4	**88.4**	0.1
ND	19.9	9.6	7.8	1.2	**61.4**

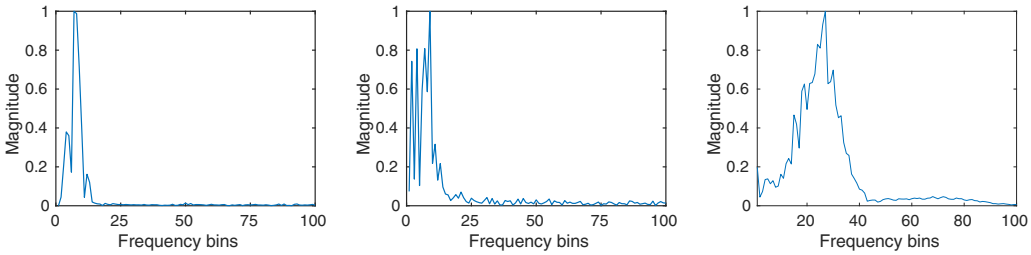
(d) Diffraction waves.

%	LF	LP	SI	PO	ND
LF	**52.3**	10.4	14.5	5.6	12.1
LP	11.1	**40.4**	15.7	19.6	13.2
SI	27.4	14.8	**36.3**	13	8.4
PO	8.2	16.6	11.4	**32.5**	31.3
ND	18.9	17.2	10.4	21.9	**31.5**

(e) Lateral and Backwall waves.

%	LF	LP	SI	PO	ND
LF	**60**	15.7	4.69	2.2	17.3
LP	19.5	**74.7**	2.5	1.3	1.9
SI	3.1	2.4	**72.5**	17.5	4.5
PO	0.4	0	8.5	**90.6**	0.0
ND	5	10.3	2.9	1.2	**80.6**

backwall wave and entire ToFD signal. Table 5.6 shows the features (spectral content) required for each case in Figure 5.7 and the respective average and maximum accuracy performance. Since the original signal sample frequency was 500 MHz, this leads to the peak frequency (5 MHz) being localized around the 7th, 8th and 25th frequency bin (Figures 5.7a, 5.7b and 5.7c respectively). The displayed windows show the frequency bins range up to the 100th bin, as the entire ToFD signal contains frequency content below this limit. It can be seen that the segmentation approach still provides relevant content while reducing the observable frequency bins count to up to 20.



(a) Lateral wave. (b) Backwall wave (c) Entire ToFD signal

Figure 5.7: Typical magnitude spectrum comparison for Lack of Penetration class.

Table 5.6: Example of feature set reduction in Lack of Penetration class and associated accuracy.

segment	features	average accuracy (%)	max accuracy (%)
ToFD signal	90	76.9	97.5
lateral wave	up to 20	70.0	85.3
backwall wave	up to 20	68.0	88.0
lateral+backwall	up to 40	75.7	92.7

5.4 Final remarks

This Chapter presented the feasibility of employing ELM networks in the task of weld defects classification. This is a step towards embedding a classifier capable of executing the training step locally in a microcontroller, using the benefits of ELM networks for such device as discussed in Section 3.5. ELM networks is the classifier adopted in the remainder of this work.

An approach for segmentation of ToFD signals was proposed, and the results indicate that a composition of the frequency magnitude spectra of Backwall and Lateral waves is comparable to using the spectrum for the entire ToFD signal (as verified in works in the literature). Transforming the entire ToFD signal to frequency domain using Fourier analysis may even be inadequate due to the dynamic content of the signal, which is the case of the present work. The said composition of spectra will be used as input in the remainder of this work.

Chapter 6

Embedded ELM

6.1 Introduction

The outcomes of the investigation shown in Chapter 5 enable the next step in classifier implementation in the microcontroller, which is tuning the data set representation and the constituting elements of ELM networks for embedding them in the microcontroller.

The next Sections present discussion regarding data representation and feature extraction considering methods such as PCA and autoencoder (see Chapter 3), assessment of probability density functions in random weight generation and discrete random weight encoding in ELM.

6.2 Feature Extraction

Feature extraction is a convenient method for reducing the amount of memory used in an application by reducing data dimensionality and preserving relevant information while also saving computation time, as both factors are heavily dependent on the input dimensionality n.

Following the segmentation approach presented in Chapter 5, the idea is to apply feature extraction methods in the data set formed by the composition of frequency magnitude spectra of the lateral and backwall waves from each ToFD sample. The starting dimensionality (considered bins from magnitude spectra) of an input here differs slightly from the one obtained in Section 5.3 (originally 40), since

inspecting the resulting magnitude spectra for both waves, the high frequency region contains almost null energy components. Eliminating these components and the DC component from both spectra (lateral and backwall), the resulting dimensionality for an example signal is 24. This number is still not feasible for implementation in the microcontroller with the current training and test set (see Appendix A). Dimensionality reduction also results in the benefit of using less neurons at the hidden layer of the ELM network, since the mapping space provided by the hidden layer would have less information at the input layer to work around.

The selected techniques to perform dimensionality reduction were PCA and ELM-based autoencoders (ELM-AE). PCA is a popular linear method for feature extraction and has been successfully used in ultrasound NDE problems [86], whereas ELM-based autoencoders tend to produce more compacted data and would also represent code reuse, since the embedded classifier is intended to be of ELM type. The methods were compared with each other regarding classifier performance, data compaction and computational cost. Both techniques were applied to the whole data set in order to obtain their respective transformation matrices.

A practical dimensionality limit must be adopted for feature extraction. In order to work with a known measure of dimensionality reduction, this procedure was based on inspection of components provided by applying PCA to the data set, since it is possible to estimate the percentage of information retained in a given number of coefficients (variance, in PCA's case).

Figure 6.1 depicts how the PCA components for the combination of magnitude spectra form lateral and backwall waves of this data set relate to its total variance. It has been noticed that:

- By using the first 10 components, 90% of variance is achieved;
- Increasing the coefficients count to 14 produces 96% of total variance.

Hence, the last 10 coefficients are responsible, together for only 4% of total variance. Therefore, 14 coefficients will be the upper limit for dimensionality reduction considered in this work for both PCA and ELM-AE, since it comprises a great share (96%) of variance for PCA for reasonably few coefficients. In general, ELM-AE might produce greater compaction performance than PCA because of its nonlinear property.

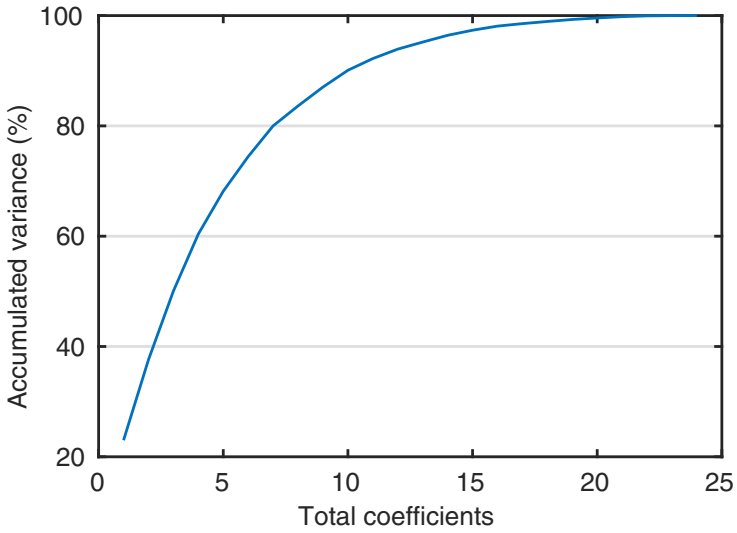


Figure 6.1: PCA load-curve for the considered data.

Figures 6.2 to 6.4 depict boxplots representing the performance of ELM classifiers trained from PCA and ELM-AE transformed inputs. Table 6.1 summarizes the results of these tests.

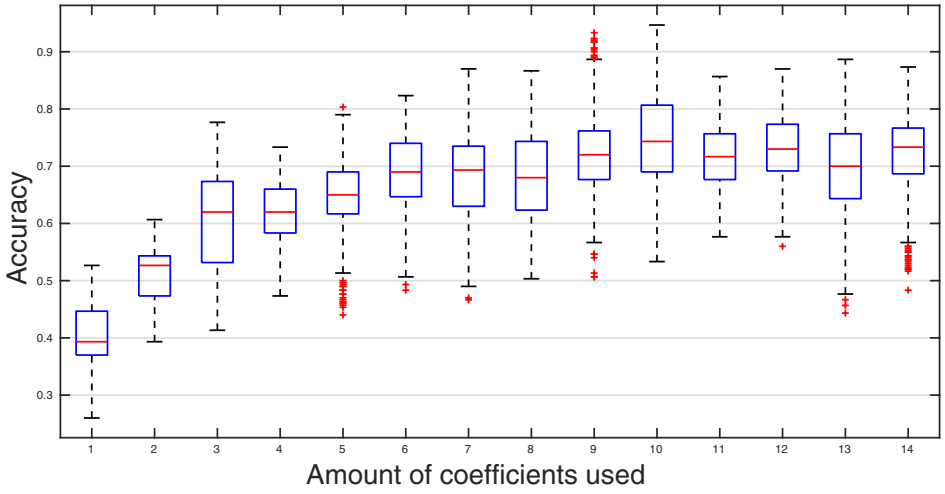


Figure 6.2: Classification performance results for PCA pre-processing.

Training sets based on PCA data (Figure 6.2) presented median accuracy around 70% with at least 6 coefficients. The higher median and maximum accuracy was achieved with 10 coefficients. Linear ELM-AE (Figure 6.3) presents slightly lower performance than PCA, and produces comparable results with PCA around 9 coefficients (hidden neurons), having its best result at 14 coefficients. On the other hand, Nonlinear ELM-AE presented more variance in accuracy performance, and

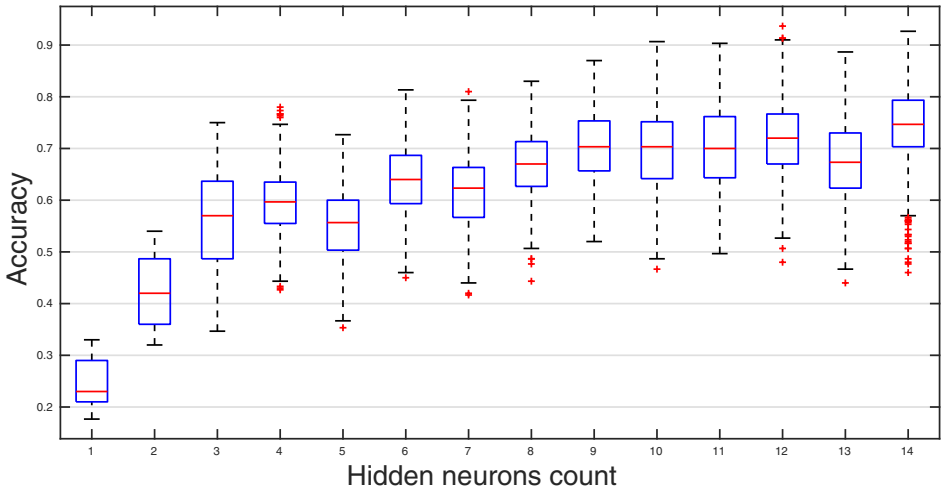


Figure 6.3: Performance results for linear ELM-AE.

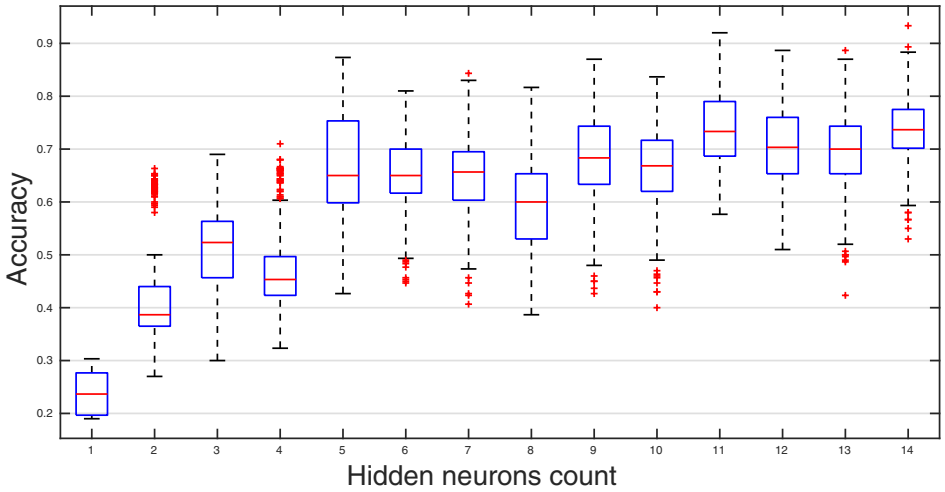
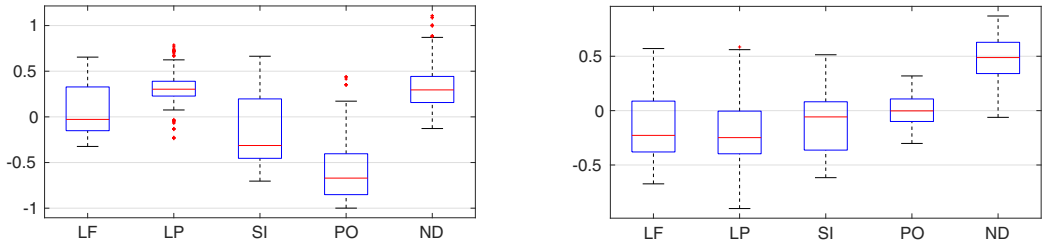


Figure 6.4: Performance results for nonlinear ELM-AE.

Table 6.1: Tested methods and best performances in feature extraction.

Method	**inputs**	**average**	**max**
PCA	10	0.746±0.082	0.947
linear ELM-AE	14	0.742±0.077	0.927
nonlinear ELM-AE	11	0.739±0.068	0.920

the best case, as shown in Figure 6.4 for 11 coefficients, is comparable with the best linear case. Looking at Table 6.1 and comparing linear and nonlinear ELM-AE, using less coefficients was considered to be more advantageous despite the slightly



(a) Boxplots of the first component of the five classes in PCA representation.

(b) Boxplots of the second component of the five classes in PCA representation.

Figure 6.5: Boxplots for first and second PCA components

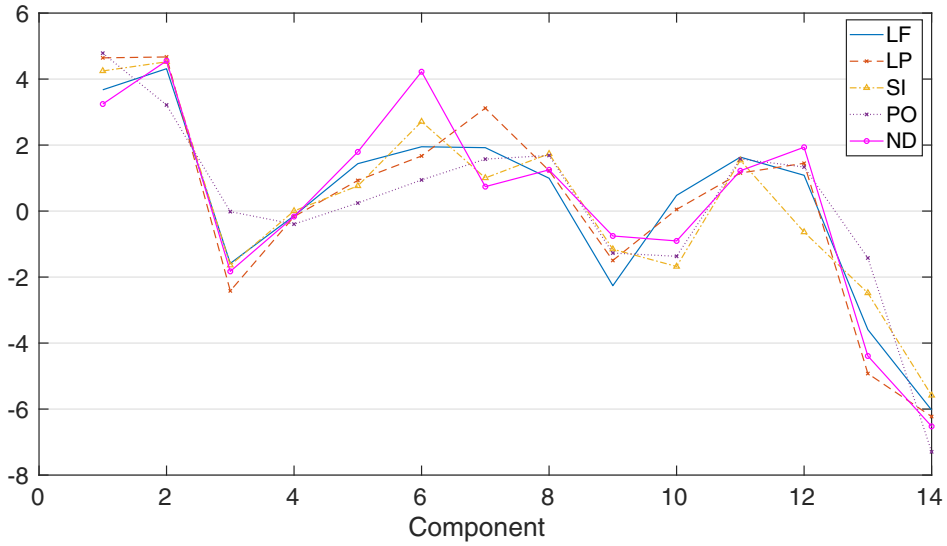


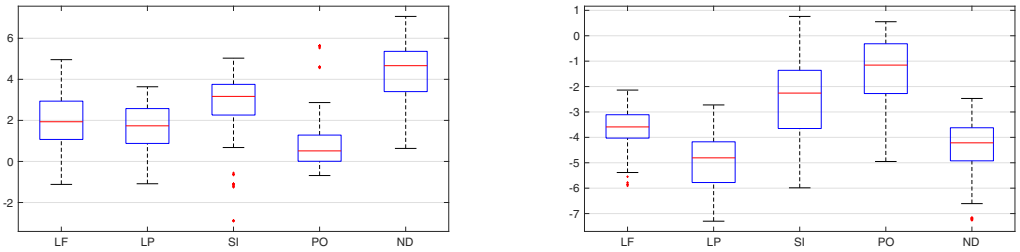
Figure 6.6: Average component representation of the five classes in ELM-AE representation.

lower accuracy performance for the nonlinear case, while neither provided better performance than PCA. Thus, further analysis in this Chapter will consider only PCA and Nonlinear ELM-AE (referred to as just ELM-AE).

Figures 6.5a and 6.5b show the boxplots for the first two components of PCA-transformed magnitude data for all five classes for all signal examples in the data set. The first component provides relatively good separation between the classes PO, LF, LP and ND, whereas the second component may contribute to eliminate the ambiguity present for LP and ND in the first component.

Since there is no contribution order rule for ELM-AE as there is for PCA, Figure 6.6 shows the average values for components of ELM-AE-transformed data for all five

classes. In this representation containing 14 components, the sixth and thirteenth component provide the wider separation between classes. Figures 6.7a and 6.7b show the boxplots for these components in the data set. Class separation in the sixth component is compromised by the classes LF and LP. The thirteenth component has more distinct class medians and dispersion, however SI overlaps PO, as it is also the case with LF and ND. Therefore, PCA data can be better separated in their two most variant components than ELM-AE data.



(a) Boxplots of the sixth component of the five classes in ELM-AE representation.

(b) Boxplots of the thirteenth component of the five classes in ELM-AE representation.

Figure 6.7: Boxplots for the two most separated components of ELM-AE representation

6.3 Random Weight Initialization and Encoding

In order to produce an embedded decision support system capable of executing the training step *in situ*, the main issues to address are memory usage and processing performance, usually related to time spent in calculations. Data dimensionality reduction, as discussed in the previous Section, is a good starting point in improving memory usage.

The ELM algorithm greatly improves the processing performance by executing the training step analytically (Equations 3.29 and 3.30). Given the reduced resource paradigm present in microcontrollers, iterative approaches with intensive calculation such as the backpropagation algorithm would lead to practical issues in the microcontroller context by stacking up time spent in calculations and time needed for iterations. From the memory usage point of view, another advantage of the ELM algorithm is the fact that the random weights don't need to be updated, keeping the

same values as they were initialized. Based on this fact, newer training procedures in the same system may use the same set of random weights, provided they have been previously stored in non-volatile memory.

Even following the pre-storing approach for random weights, the network size may still lead to an inconvenient task of having possibly thousands of random weights listed in source files, which possibly diminishes code maintainability and forces memory trace according to the implementation (fixed point, single or double precision variables). For instance, the 60-neuron hidden layer and the 40-node input network tested in Chapter 5 results in $n \times L = 2400$ random weights.

In this work, the proposed approach to overcome this issue is to encode the random weights based on an evaluated bit count B (as opposed to the work in [88] where weights are indeed binary values), resulting in a set of 2^B quantized random weights $w_0, w_1, \ldots, w_{2^B-1}$. This way, instead of coding each and every random weight into memory, the weights are stored under the form of an array of 32-bit random integers. The random weights are retrieved by sweeping the integers array by steps of length B. The encoding procedure occurs as follows:

1. Establish an amount of bits B. This is the length of the bit word in which the weights will be encoded.

2. Store an array of 32-bit random integers.

3. Sweep the 32-bit integer array with a logic AND operation between a section of the 32-bit integer and a bit mask of 1s whose length is B.

4. Retrieve the random weight by using a lookup table matching the obtained bit word and the respective random weight index in the discrete random weight set.

Figure 6.8 shows an example of the above procedure using $B = 3$. A random weight set $w_0, w_1, \ldots, w_7$ has been previously determined. The first logic AND operation between the bit word and the first block of $B = 3$ bits resulted in the bit word 110, corresponding to weight index "6", thus informing that the random weight to be used is w_6. The process continues until the needed number of random weights has been retrieved from the random 32-bit integer array,

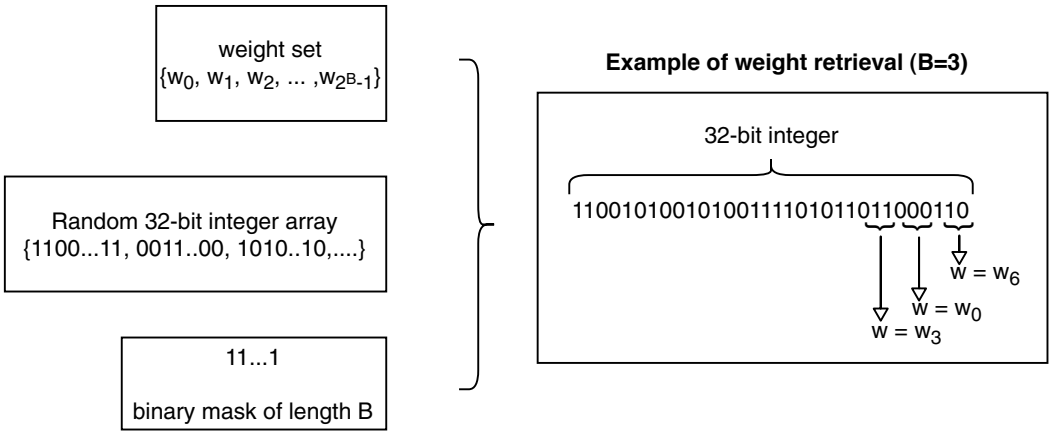


Figure 6.8: Example of random weight decoding.

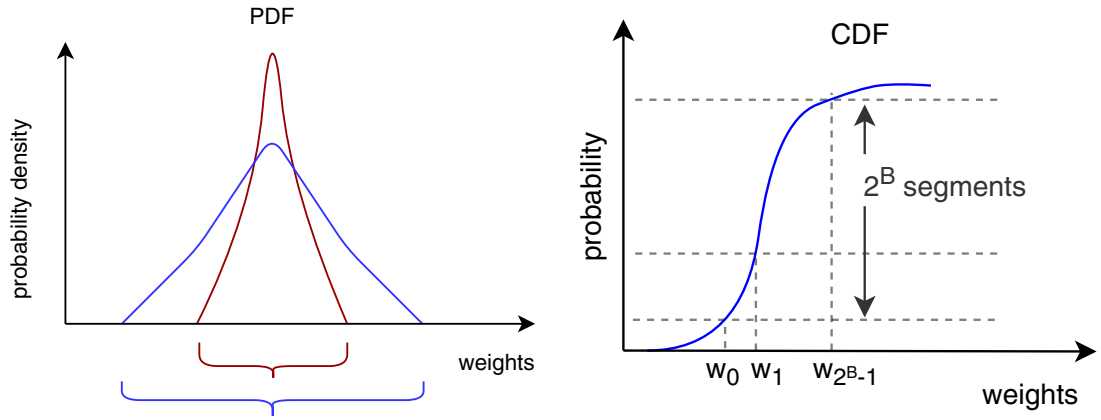
A single 32-bit integer may contain up to $32/B$ coded weights, rounded down. Hence, for a network containing $n \times L = W$ weights, the length l for a 32-bit integer array is the least integer number that satisfies

$$l = \frac{W}{\frac{32}{B}} = \frac{WB}{32} \tag{6.1}$$

The binary encoding approach for random weights replaces the theoretical infinite possible random values (which are in fact limited by format precision) assigned to a weight for a finite and considerably smaller set of weights. This new set of weights must have its values chosen according to a determined domain in the respective probability density functions. The following listed method, supported by Figure 6.9, is proposed for a given PDF in order to obtain a set of 2^B discrete weights:

- Determine the domain interval in the generating probability density function. This implies in how stretch is the envelope of the function and thus how much variance is inherent to the model. For infinite domain functions, a practical limit must be adopted (see Figure 6.9a).

- Determine which values to assign to each weight in the set of random weights. An approach for obtaining such values is to look at the Cumulative Distribution Function (CDF) of the respective PDF and inspect which values in the function's domain divide the CDF equally in 2^B parts (Figure 6.9b). In other words, which points in the domain divide the region beneath the PDF



(a) Illustration of domain evaluation in a generic PDF.

(b) Illustration of weight quantization in a generic CDF.

Figure 6.9: Steps in methodology for obtaining discrete random weights.

envelope in equal area sections. Once determined the set of discrete weights, when instantiating a new random weight for the network, it will assume the value of the nearest discrete weight in the set.

This procedure tries to leverage the use of discrete weights by balancing the occurrence of the discrete weights in order to provide a better representation of the PDF, specially for low values of B.

.

Having been defined the procedure for weight encoding, the next step is to determine which random weight generating context is better suited for this task.

6.3.1 PDF Analysis

In this work, the random weight generator PDF was also a subject of study, since it may influence the performance of the considered problem [56].

Four distinct probability density functions where evaluated in this work: Exponential, Normal (Gaussian), Uniform and Weibull. These PDFs were selected according to their symmetry around their mean values, comprising two different symmetric (Normal and Uniform) and two different asymmetric (Exponential and Weibull) PDFs.

- **Exponential**. This function has nonzero values for positive domain. It is

determined by Equation 6.2:

$$f(x) = \begin{cases} \Lambda e^{-\Lambda x}, & x \geq 0 \\ 0, & x < 0 \end{cases} \tag{6.2}$$

This PDF has its mean equal to its standard deviation $\frac{1}{\Lambda}$. In this work, this function's practical interval length was assumed as 5 standard deviations.

- **Normal (Gaussian)**. This function may be evaluated for positive and negative values. It is defined by Equation 6.3

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \tag{6.3}$$

where μ is its mean and σ is its standard deviation. The domain for this function in this work was limited between $\mu - 3\sigma$ and $\mu + 3\sigma$, corresponding to 99% of the area beneath the PDF envelope.

- **Uniform**. This function is constant for a defined finite interval. It is defined by Equation 6.4

$$f(x) = \begin{cases} \frac{1}{2c}, & \nu - c \leq x \leq \nu + c \\ 0, & x < \nu - c \quad or \quad x > \nu + c \end{cases} \tag{6.4}$$

where ν is its mean and c is an arbitrary constant.

- **Weibull**. The Weibull PDF has nonzero value for positive domain. Its envelope is determined by two factors, as seen in Equation 6.5

$$f(x) = \begin{cases} \frac{k}{\gamma}\left(\frac{x}{\gamma}\right)^{k-1} e^{-\left(\frac{x}{\gamma}\right)^k}, & x \geq 0 \\ 0, & x < 0 \end{cases} \tag{6.5}$$

where $k > 0$ and $\gamma > 0$ are referred to as shape parameter and scale parameter, respectively. Its mean and standard deviation are not readily obtained form its equation and must otherwise be calculated. This function‘s domain was limited to 5 standard deviations.

All functions had their mean values set to 0, so that positive and negative random weights could be generated, in order to avoid saturation of the neurons' activation function in case inputs with the same signal are presented to the classifier.

In order to investigate whether there would be a proper domain to generate random weights in each PDF which would produce higher accuracy performance, all the considered functions were evaluated according to a varying characteristic parameter: standard deviation for Exponential ($\frac{1}{\Lambda}$, see Figure 6.10) and Normal (σ, see Figure 6.11), range interval for Uniform (Figure 6.12) and Shape (k) for Weibull (Figure 6.13). These parameters are related to the variance of each PDF.

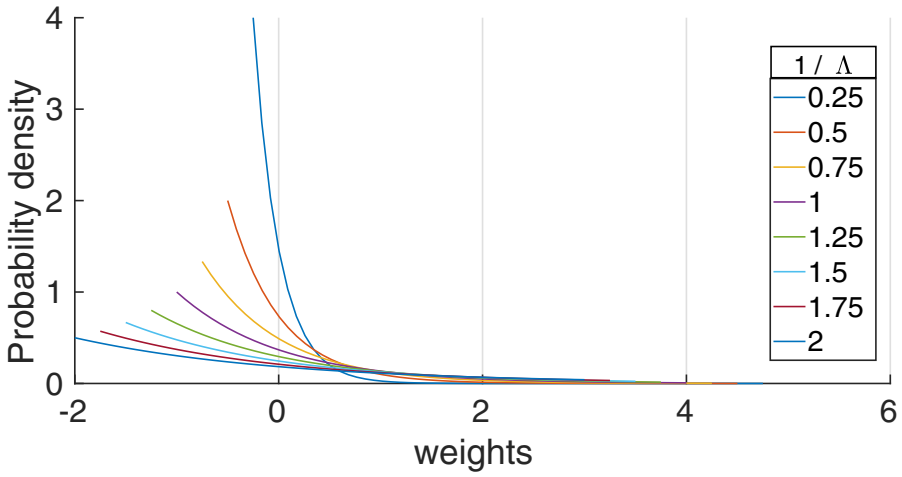


Figure 6.10: Inspected standard deviations for Exponential PDF

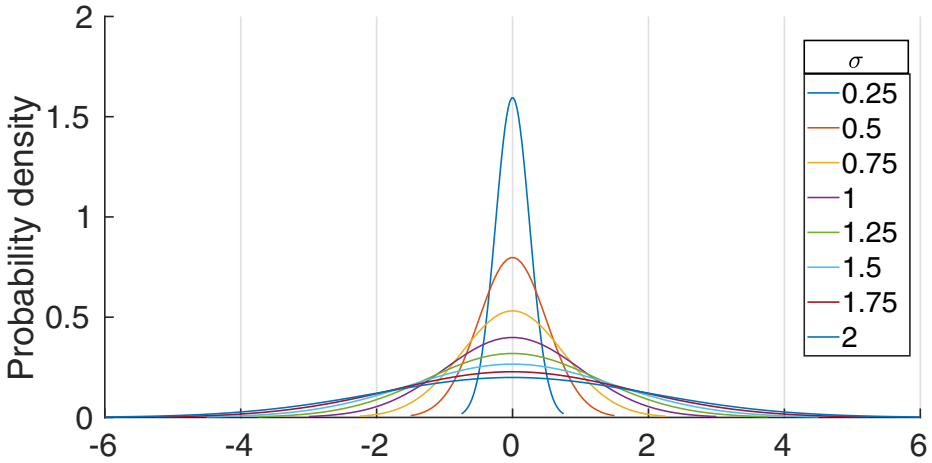


Figure 6.11: Inspected standard deviations for Normal PDF.

All tests for performance of probability density functions and binary encoding of random weights were executed using ELMs containing 20 hidden neurons. This number was found to be of good memory compromise in embedding ELMs in the microcontroller in the present work.

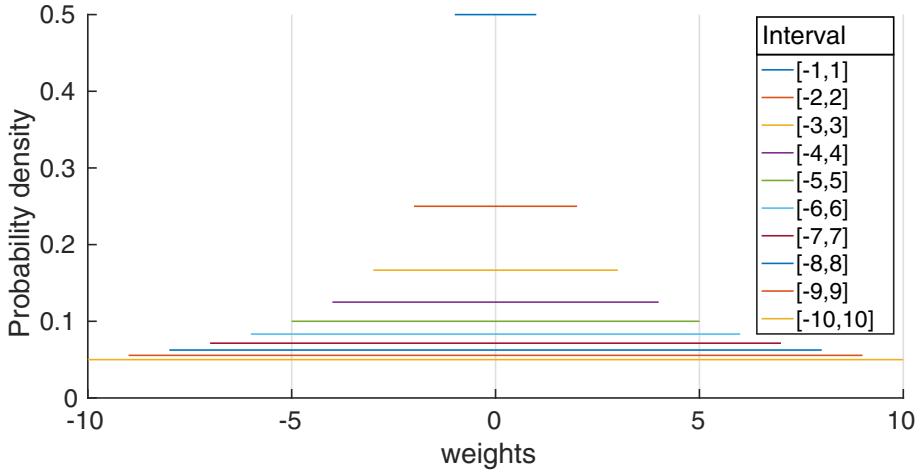


Figure 6.12: Inspected intervals for Uniform PDF.

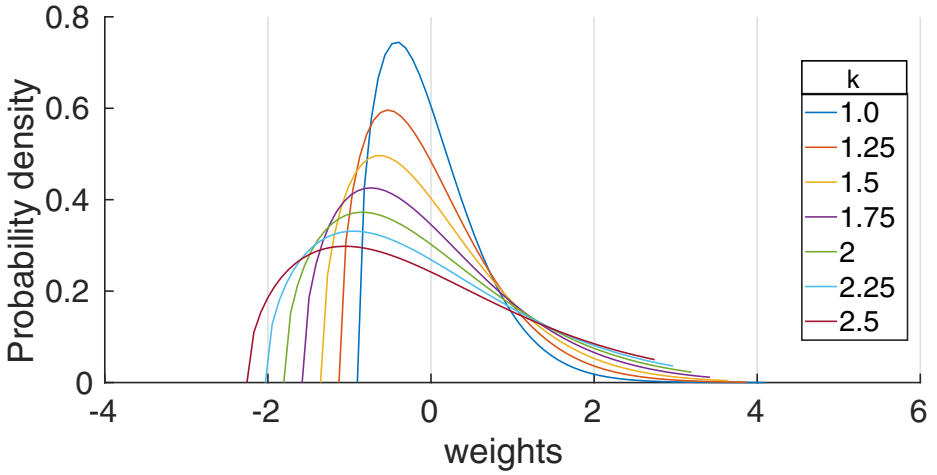


Figure 6.13: Inspected shape parameters for Weibull PDF.

Figures 6.14 to 6.21 show the resulting average accuracy surfaces from tests with networks whose random weights were generated according to each PDF depicted in Figures 6.10 to 6.13. These surfaces were obtained by varying the respective PDF parameter and also the network's regularization constant C (see Equations 3.29 and 3.30), in order to check whether this constant influences the network's performance, as seen in [51]. The performance analysis was evaluated considering networks trained and tested using (i) PCA inputs and (ii) ELM-AE inputs. Table 6.2 contains the five best cases, arranged in descending order of average accuracy and also featuring information about best accuracy reached in each test and values for the respective studied parameter and the regularization constant C, represented as $\log_2(\mathrm{C})$ for the sake of compaction.

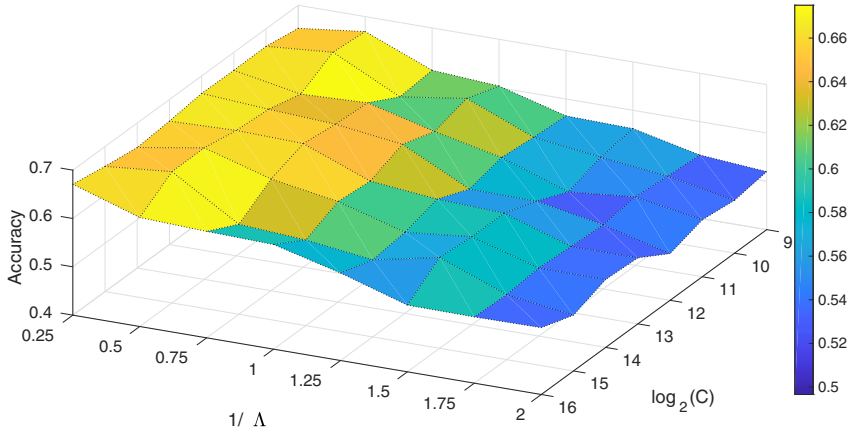


Figure 6.14: Average accuracy surface for Exponential PDF - ELM-AE.

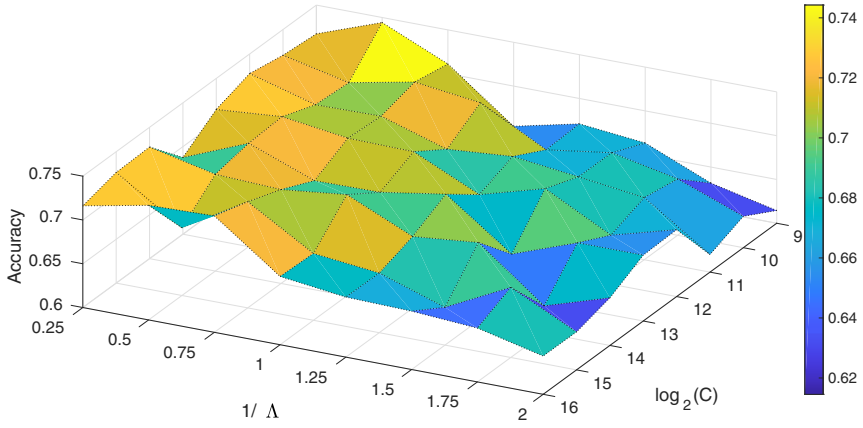


Figure 6.15: Average accuracy surface for Exponential PDF - PCA.

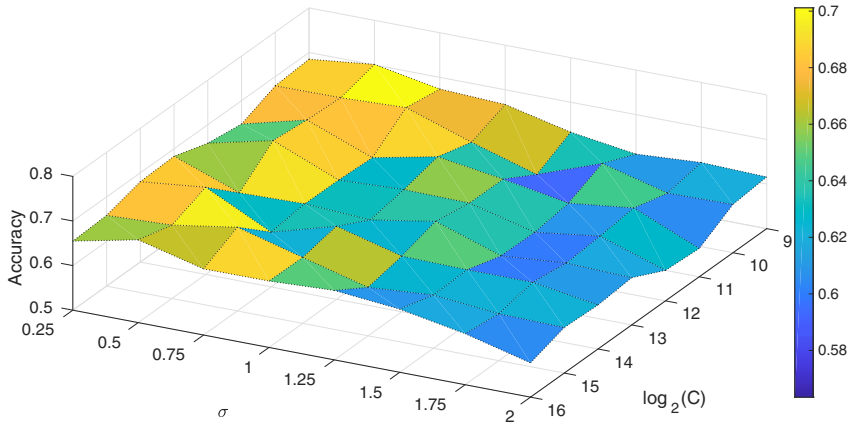


Figure 6.16: Average accuracy surface Normal PDF - ELM-AE.

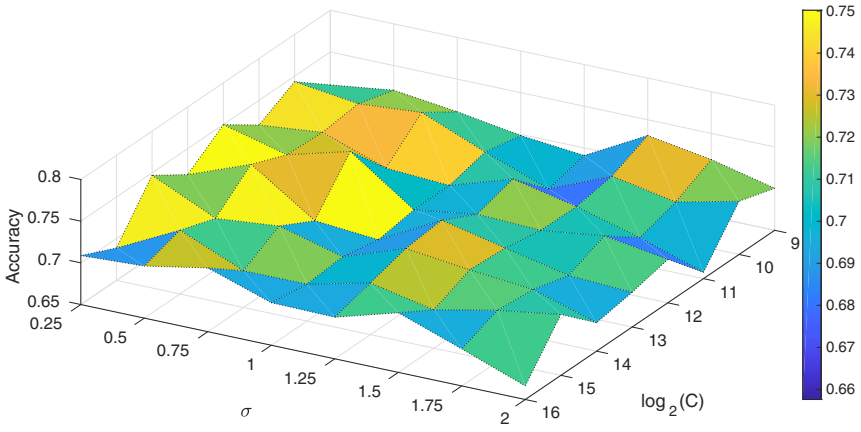


Figure 6.17: Average accuracy surface Normal PDF - PCA.

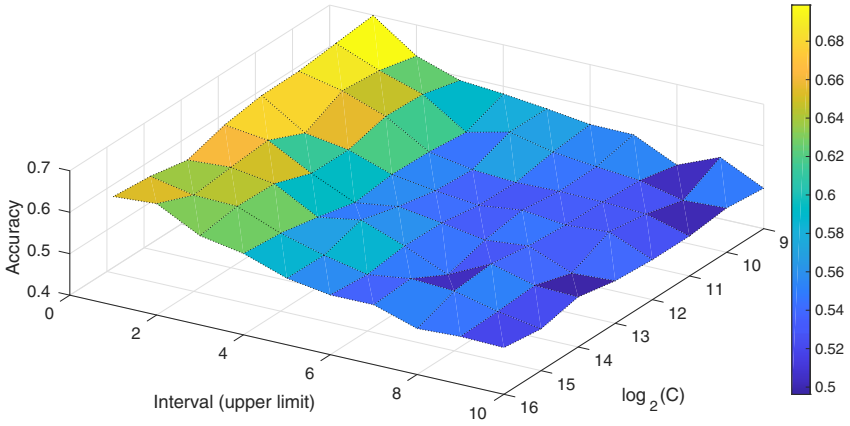


Figure 6.18: Average accuracy surface for Uniform PDF - ELM-AE.

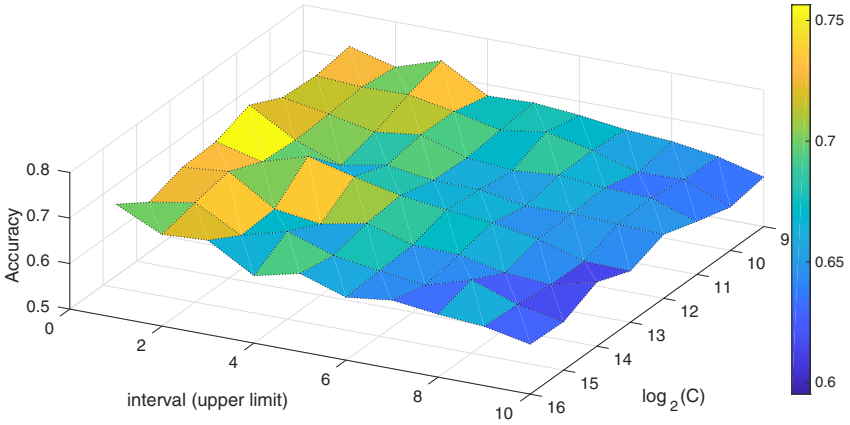


Figure 6.19: Average accuracy surface for Uniform PDF - PCA.

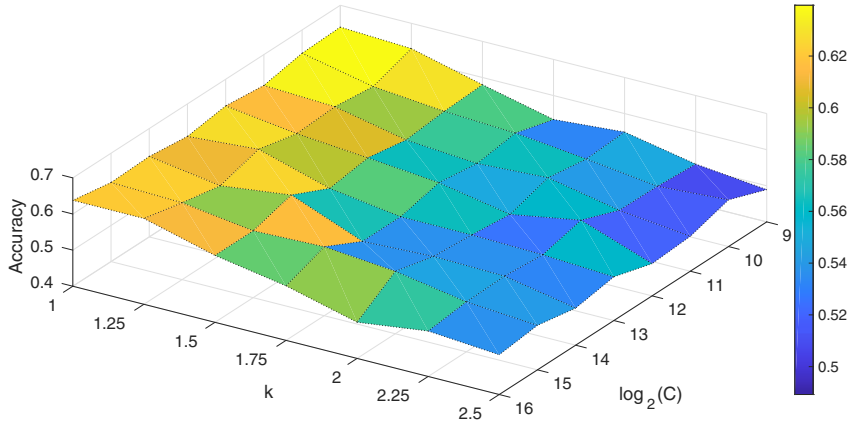


Figure 6.20: Average accuracy surface for Weibull PDF - ELM-AE.

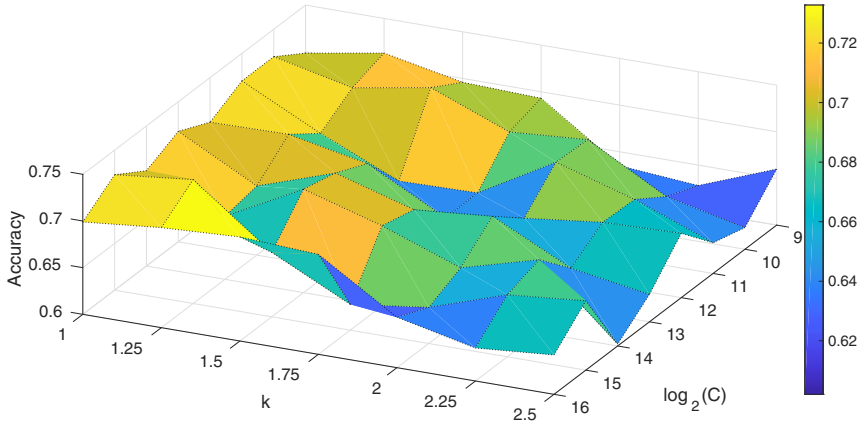


Figure 6.21: Average accuracy surface for Weibull PDF - PCA.

For all evaluated PDFs, the results for each test presented the same trend for certain aspects:

- Considering the varying parameter in all PDFs it was observed that greater average accuracies were achieved in the cases where the function domain is more compact (and therefore presents lower variance). This means that random weights generated by PDFs with sparser domain lead to a drop in generalization performance of networks in this problem.

- All PDFs presented similar average accuracy in their best cases, being Normal and Uniform the ones with slightly better performances among all.

- Symmetry does not seem to be a relevant factor in random weight generation

Table 6.2: Best cases in PDF tests, for PCA and ELM-AE features with their respective average and max accuracies and respective values for the PDFs' evaluated parameter and network's regularization constant C.

PDF	**Average**		**Max**		**Parameter**		**$\log_2$(C)**	
	ELM-AE	**PCA**	**ELM-AE**	**PCA**	**ELM-AE**	**PCA**	**ELM-AE**	**PCA**
Exponential	0.675±0.081	0.744±0.081	0.910	0.950	0.50	0.50	10	9
	0.672±0.071	0.733±0.078	0.837	0.930	0.50	0.75	14	16
	0.672±0.086	0.731±0.066	0.853	0.900	0.50	0.50	15	16
	0.671±0.087	0.728±0.092	0.887	0.900	0.25	0.25	16	11
	0.669±0.073	0.728±0.075	0.803	0.883	0.50	0.25	9	14
Normal	0.701±0.069	0.750±0.088	0.870	0.95	0.50	0.75	9	12
	0.695±0.081	0.750±0.073	0.897	0.883	0.50	0.25	14	12
	0.692±0.081	0.748±0.062	0.870	0.923	0.50	0.50	12	13
	0.688±0.071	0.747±0.062	0.877	0.880	0.75	0.25	15	14
	0.686±0.061	0.743±0.079	0.857	0.957	0.75	0.25	10	10
Uniform	0.699±0.084	0.756±0.064	0.887	0.883	$[-1, 1]$	$[-1, 1]$	9	12
	0.689±0.084	0.749±0.063	0.880	0.897	$[-1, 1]$	$[-1, 1]$	10	16
	0.679±0.071	0.735±0.067	0.877	0.933	$[-1, 1]$	$[-2, 2]$	11	14
	0.677±0.079	0.734±0.075	0.903	0.94	$[-1, 1]$	$[-3, 3]$	12	13
	0.668±0.081	0.733±0.068	0.847	0.893	$[-2, 2]$	$[-3, 3]$	16	9
Weibull	0.640±0.061	0.733±0.064	0.810	0.880	1.00	1.25	9	15
	0.638±0.068	0.724±0.088	0.837	0.95	1.00	1.00	16	15
	0.638±0.098	0.721±0.063	0.907	0.877	1.25	1.00	16	10
	0.636±0.076	0.721±0.070	0.850	0.890	1.00	1.00	10	11
	0.630±0.069	0.718±0.060	0.807	0.870	1.25	1.00	9	13

in the average accuracy performance point of view.

- The performance for PCA inputs is in general better than the one for ELM-AE inputs in a given PDF. It indicates that, for the current data set (ultrasound signals frequency spectra of weld flaws), the nonlinear encoding feature of compact ELM-based autoencoders may pass beyond the point of mapping discriminant features than using linear remapping provided by PCA.

- The regularization factor C showed no observable trend in either the evaluated PDFs or dimensionality reduction technique. The best cases shown in Table 6.2 for each PDF show no trend for C in the studied interval.

Once the best conditions in terms of domain extension for each PDF are understood, these parameters can be used to establish the limits in which the discrete

weights may be obtained.

6.3.2 Performance of discrete random weights

The random weight encoding approach was tested in all studied functions using the best performance scenario obtained for each one. PCA and ELM-AE data were tested, in order to assess the performance for these feature extraction methods under a more restrict scenario (the random weight encoding) and compare them with the results shown for the high precision-coded random weight case.

Figures 6.22 to 6.29 show the performance boxplots regarding accuracy for each considered PDF and feature extraction method. For each test, the variables shown in Table 6.2 (columns Parameter and $\log_2(C)$) for the respective best cases were used.

The discrete weights were generated using a specific number of bits per test, starting at 1 bit and going up to 8 bits, leading respectively to 2^1 to 2^8 possible random weights for each PDF. The exception is the Normal PDF, where testing for 1 bit would be the same as the Uniform PDF since both are symmetric and therefore would produce two values of random weights with same probability. Thus, the initial test value was 2 bits. Testing up to 8 bits was considered to be a practical limit for encoding the weights in binary arrays in the embedded system.

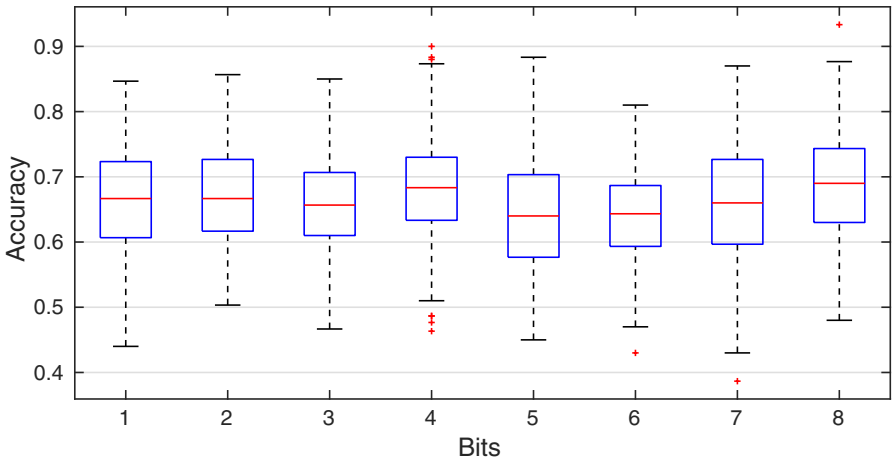


Figure 6.22: Weight quantization tests for Exponential PDF - ELM-AE.

The performance of each PDF in Figures 6.22 to 6.29 are summarized in the following points:

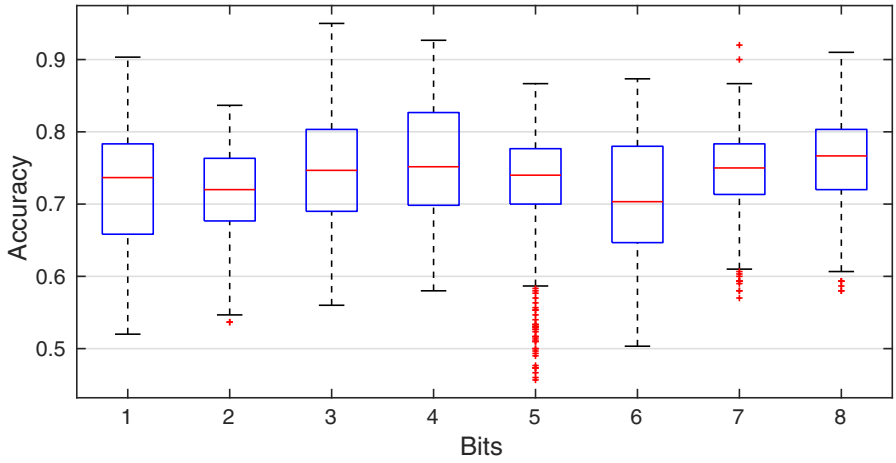


Figure 6.23: Weight quantization tests for Exponential PDF - PCA.

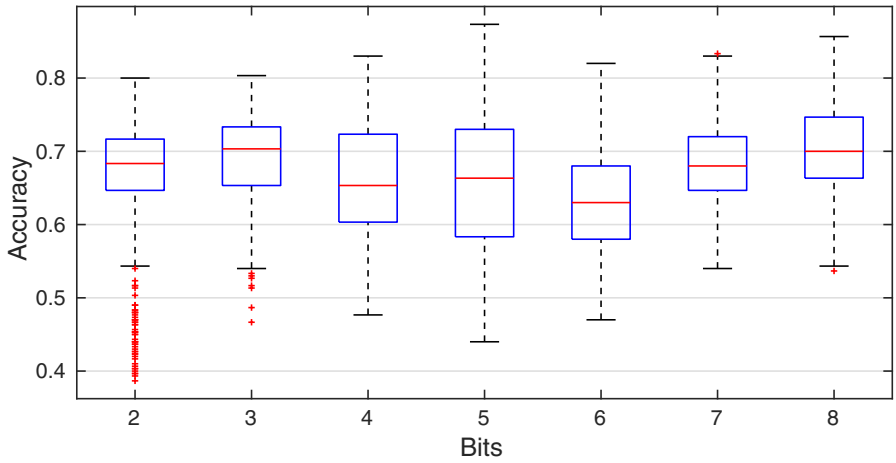


Figure 6.24: Weight quantization tests for Normal PDF - ELM-AE.

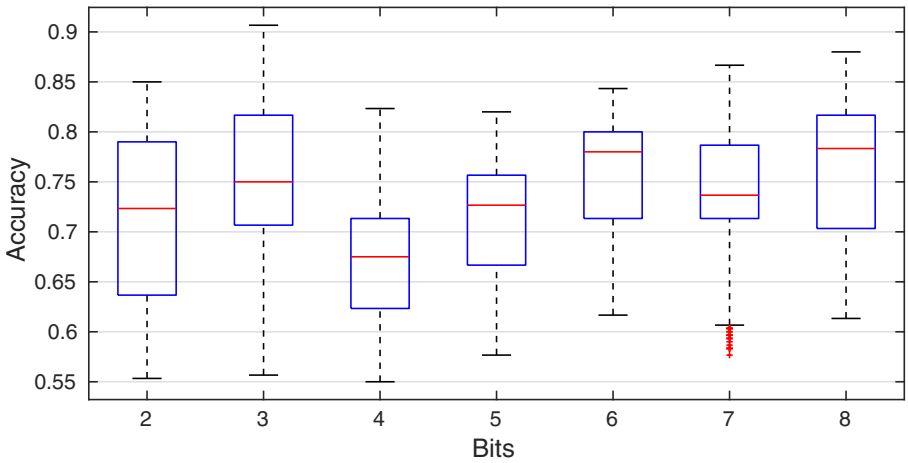


Figure 6.25: Weight quantization tests for Normal PDF - PCA.

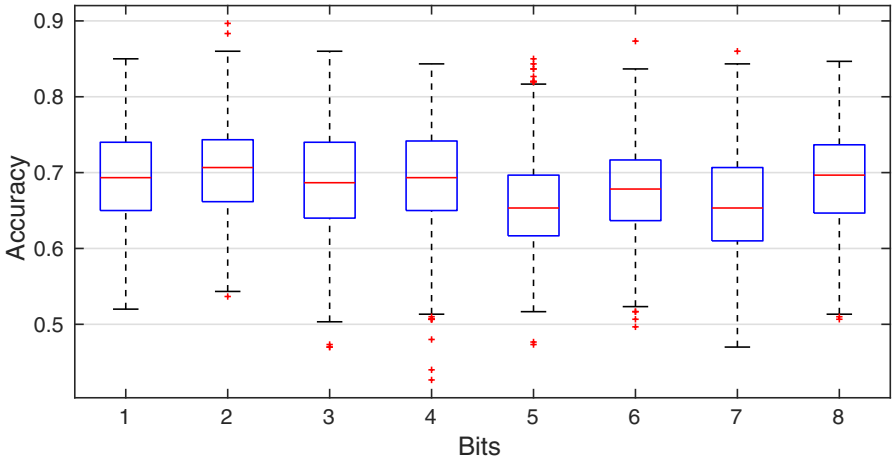


Figure 6.26: Weight quantization tests for Uniform PDF - ELM-AE.

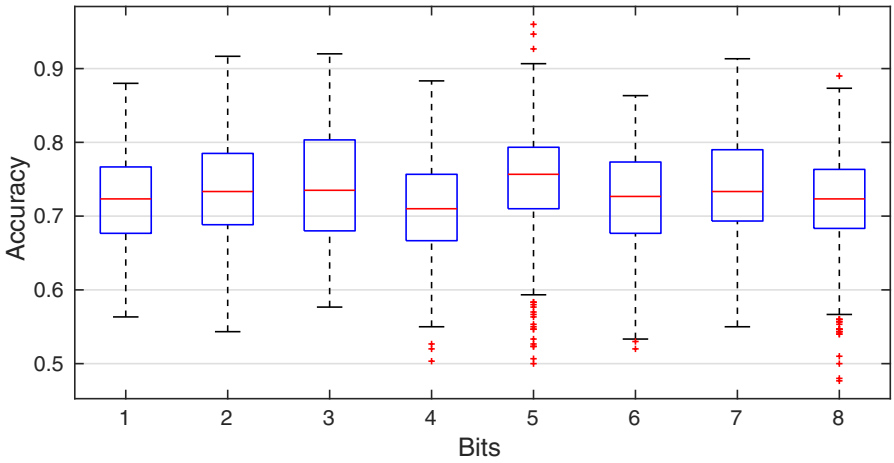


Figure 6.27: Weight quantization tests for Uniform PDF - PCA.

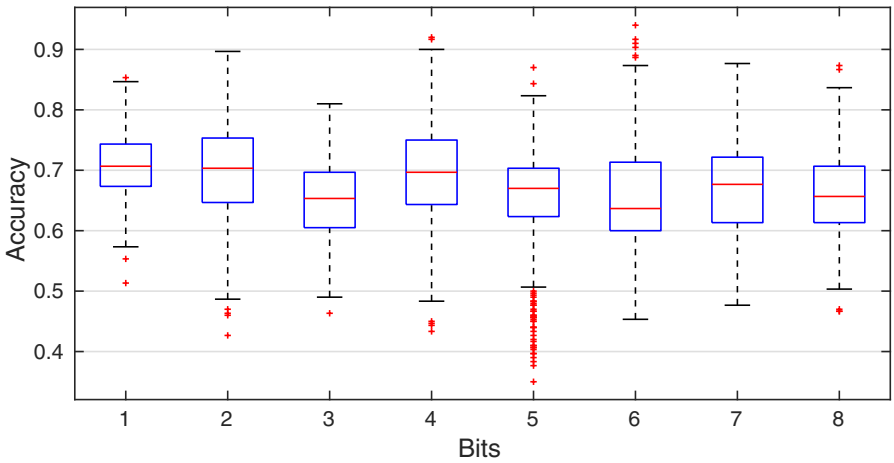


Figure 6.28: Weight quantization tests for Weibull PDF - ELM-AE.

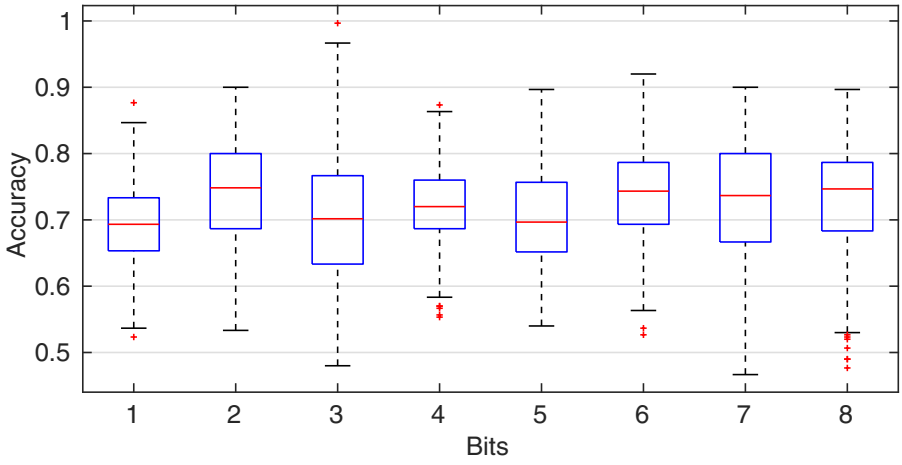


Figure 6.29: Weight quantization tests for Weibull PDF - PCA.

- The Exponential PDF presented accuracy performance concentrated in the interval between 0.6 and 0.7 for ELM-AE features, where as PCA features provide overall higher accuracy performance. Both types of features don't present a trend with the increase of bits.

- The Normal PDF also present overall better accuracy performance for PCA features than ELM-AE. Increasing the bit count in the PCA case seems to reduce accuracy dispersion while increasing accuracy performance.

- The Uniform PDF has higher accuracy performance for PCA features than for ELM-AE ones, in the same way as the other PDFs. This function produces high accuracies with low bit count and its dispersion seems to be roughly independent of the considered bit count.

- The Weibull PDF also presents higher accuracy performance for PCA features than for ELM-AE ones. The medians and dispersion vary throughout the considered bit count with no observable trend. The results for 3 bits spans from almost perfect accuracy to one of the worst registered ones.

The results for accuracy performance are consistent with the ones shown in the average accuracy surfaces and data in Table 6.2, where PCA data presents general better accuracy performance when compared with ELM-AE data. ELM-AE was tested in order to investigate the performance of its compaction capability and code reuse (since it is, also an ELM-based method, like the classifier solution in this work),

however its discriminant performance is inferior when compared to PCA. The gain accuracy performance for PCA inputs is compensated by extra computational effort, which may be achieved by another dedicated microcontrolled unit in the decision support system.

Since the goal is to maximize accuracy performance with low bit count for random weight generation, Exponential and Uniform functions show the best performance in this regard, with medians around 0.75 and maximum values above 0.9 while presenting less dispersion, matching the Lateral+Backwall profile in Figure 5.6.

6.4 Tests in microcontroller

As a validation of the previous exposed approach, three experiments were executed in the microcontroller, comprising the training and test step, by using as an example a set of discrete random weights encoded using 1, 2 and 3-bit lengths generated using the Uniform PDF within the range [-1, 1]. This is the simplest function to implement weight decoding (retrieve weight values from binary array) and it stands as one of the best accuracy performance enablers among the studied functions, being capable of reaching high accuracies while having a more predictable dispersion than the other distributions. Therefore, it was selected for demonstration in the microcontroller.

The results in the previous Section show that networks working with PCA-transformed inputs provide better accuracy performances than networks trained to deal with ELM-AE-transformed data. For this reason, for the tests in the microcontroller presented in the current Section, all networks were trained and tested using PCA-transformed data. Since the random weights are fixed in memory for the reasons presented in Section 3.5, the procedure for tests described in Section 4.5 was applied in the microcontroller tests skipping step 3 and using a total amount of 50 redefinitions in step 4.

The results for a 20-hidden neuron network is shown in Figure 6.30. whose boxplots in Figure 6.30 show accuracy performances compatible with the previous results showed in Figure 6.27, having the 1- and 3-bit encoding performed similarly to the simulation results whereas 2-bit encoding did not reach accuracy performance as high as in the simulated case. The fixed random weights, in contrast to the 50

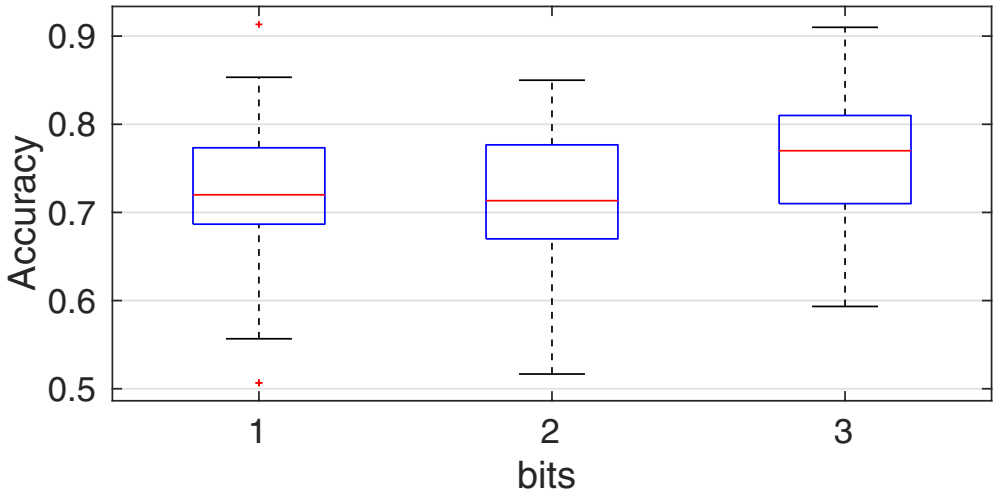


Figure 6.30: Boxplots for tests with different bit encoding.

network initializations in simulated environment (refer to Section 4.5), may be one cause behind the apparent decrease in accuracy performance for the 2-bit case.

Although the random weights have fixed values, their positions (assigned connections between inputs and hidden neurons) in the network can be easily swapped, resulting in a virtual new initialization of random weights in the network, while preserving memory usage. This procedure may be adopted in order to avoid training sub-optimal networks due to unfavorable pairings of input samples and random weights.

Nevertheless, the elapsed time for the training step was 0.314 ms and data memory usage was 94920 bytes in the current implementation. These numbers are considered good performances in the embedded platform for a training set containing 700 samples.

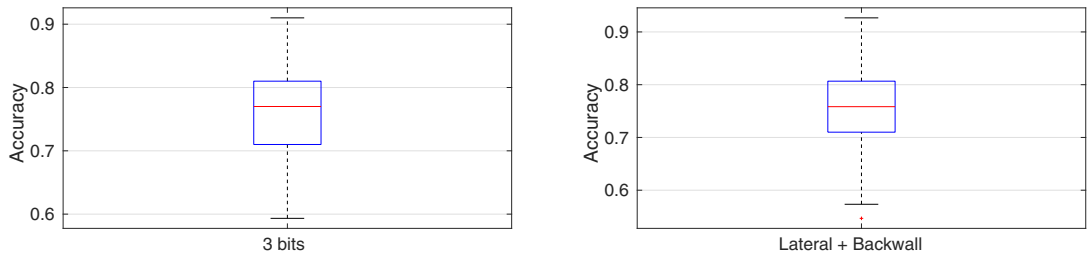
Table 6.3 presents the average confusion matrix corresponding to the test case using 3-bit coded random weights, which is considered the best case in Figure 6.30. The main diagonal presents the accuracy for each class, the rows contain a reference class and the columns contain the classification errors for each reference class. The higher average accuracy percentages are found in the classes PO and ND. The average misclassification percentage of a defect as No defect was under 8% for LF and 5% SI (both also present the lower accuracy among the classes) whereas for LP and PO the average accuracy error was below 4% and 1%, respectively.

Ultimately, despite the slightly lower accuracy performance for the 1- and 2- bit

Table 6.3: Average confusion matrix, in percentage, of the tests with PCA-ultrasound samples using 3-bit coded random weights in the microcontroller.

%	LF	LP	SI	PO	ND
LF	**70.3**	15.6	5.8	0.8	7.6
LP	16.3	**74.6**	3.3	1.9	3.8
SI	8.5	4.4	**70.6**	11.6	4.9
PO	0.0	0.4	13.6	**85.8**	0.2
ND	6.8	3.1	3.8	0.0	**86.3**

cases, these results are compatible with the ones presented for the Backwall+Lateral case in Figure 5.6, specially for the 3-bit case. As seen in Figure 6.31, the 3-bit case presented accuracy performance similar to the segmented analysis showed in Chapter 5, which in turn presented better performance than works in the literature (as discussed in Section 5.3) with the advantage of achieving dimensionality reduction for the data set and memory optimization. The proposed approach presented up to this Chapter provides evidence that it can produce decision support system in a microcontroller for classification of weld defects in weld beads with about the same performance of works in the literature.



(a) 3-bit case.

(b) Lateral + backwall (Figure 5.6)

Figure 6.31: Boxplots of accuracy performance for segmentation and discrete random weight approach.

6.5 Final remarks

This Chapter introduced an approach for embedding extreme learning machines in a microcontroller, enabling it to execute the training step locally using data from weld

defects. PCA and ELM-AE methods were evaluated as dimensionality reduction tools, with PCA providing overall performance greater then the autoencoder.

Four probability density functions used for generating random weights were evaluated, and each had the classifier's accuracy performance decreased as the PDFs' variances grow larger. There was no evident value for the regularization constant C that led to better accuracy performance in any studied function.

The discretization of random weights by bit encoding provided similar results compared to the ones obtained using continuously random generated weights. Finally, the implementation on the microcontroller provided performances closer to the simulated cases and the results in previous work in the literature.

Details on the developed ELM application for microcontroller used in the tests are described in Appendix A.

Chapter 7

Conclusions

7.1 Work review

This work presented an approach for embedding a decision support system in microcontrollers, in order to classify samples originated from a weld bead region uniting steel plates. The context of NDE decision support systems presented in Chapters 1 and 2 shows that the works developed in this area depend on personal computers to carry out the classifier training, whereas the current proposed implementation can execute the training step in microcontrollers. Moreover, Chapter 3 indicates that most works regarding classifiers in microcontrollers rely on the training step being executed in personal computers.

Chapter 4 presented the conditions in which the data set for this work was obtained as wells as the methods and metrics to evaluate the classifier performance, establishing the basis for analysing the approach presented in this work for the decision support system designed for weld defects. The obtained results were detailed in Chapters 5 and 6.

Under the scope of the work defined in Section 1.2, the main goal of this work was achieved, since the embedded ELM classifier was capable to reproduce similar accuracy performance to reported works in the literature. The specific objectives of that Section are rewritten below with the respective ways in how they were addressed:

- **Feature Selection**. The relevant parts of the ToFD magnitude frequency spectrum for classification task were used to form a sample. Moreover, a

feature extraction method was applied in order to further reduce data dimensionality.

- **Artificial Intelligence**. ELM theory was used as classifier and tuned for leveraging the available hardware resources.

- **Embedded platform**. An ELM library was developed for the selected family of microcontrollers. The platform is capable of executing the training step itself with a portable implementation.

7.2 Contributions

This work presented some contributions to the fields of Ultrasound NDE, ELM and Embedded Systems. These are listed below:

1. Introduce ELM theory in ultrasonic NDE problems regarding classification. ANN classifiers based on backpropagation training are dominant in this field, and ELM provides similar accuracy performance. The comparison between ELM and backpropagation-trained networks was published in a conference paper [89] (available only in portuguese, see also Appendix B).

2. Inspect the nature of magnitude frequency spectrum of a typical ToFD sample for the studied weld defect classes. This assessment led to elimination of irrelevant information in the ToFD signal and a new approach for dealing with frequency spectrum, based on segmentation of ToFD signals, which is different from the reported cases in the literature. This content (most part of Chapter 5) was compiled and published in journal paper [90] (see also Appendix B).

3. Evaluate feature extraction methods in the frequency content of weld defects. This investigation showed that PCA produces better accuracy performances than ELM-AE and may be employed in feature extraction for the current problem.

4. Evaluate probability density functions in ELM classifiers. This work showed in Chapter 6 that random weights generated by all studied functions provide similar performance to the network. The regularization constant C in ELM

solution plays no defining role in the studied PDFs. On the other hand, it has been shown the behavior of accuracy performance according to how sparse (in a dispersion sense) is the domain of the PDF. It is generally preferable to employ smaller domain instances of the PDFs for the current problem.

5. Random weight discretization and encoding. In order to easily embed ELMs in microcontrollers, this work presented the approach for encoding continuous random weights in bit words, so that a finite set of random weights is used to connect the hidden layer to the input layer in embedded ELMs. This procedure has a practical appeal as it eases coding and maintenance of ELMs in microcontrollers while enables memory reuse.

6. Embedded classifier capable of local (in situ) training. This is the result of the combination of the previous points and features an approach for embedding classifiers in microcontrollers. This item along with items 3, 4 and 5 (corresponding to Chapter 6 were summarized into a submitted journal paper, currently under revision (See Appendix B). The coded algorithms and libraries developed for this work are open to the scientific and engineering world community. The library built for this work is an open source project. It can be download, forked and shared by the scientific and engineering community at `https://github.com/lucascsilva/elm_mcu`.

7.3 Suggestions for future works

As a closing Section for this text, there are a few opportunities for further investigation related to some aspects of classification of ultrasound NDE samples and embedding ELMs in microcontrollers that may be tackled in future developments. These suggestions are listed below:

- Verify whether adding phase spectrum information is relevant for accuracy performance and its trade-off with increasing the input dimensionality.

- Extend the probability density function assessment and discrete random weight approach shown in this work for other classification problems.

- Develop ELM variants for the embedded application designed for this study, such as Online Sequential ELM (OS-ELM, refer to sub-Section 3.5.1).
- Investigate alternative methods to PCA as feature extraction approach.
- Integrate the current classifier implementation with modules for ultrasound generation and acquisition, resulting in a hardware capable of executing measurements and decision support.
- Use of other fast training architectures for classifiers, such as Echo State Networks (ESN).

Bibliography

[1] HELLIER, C. J., *Handbook of Nondestructive Evaluation*. McGraw-Hill, 2001.

[2] CARVALHO, A. A., REBELLO, J. M. A., SOUZA, M. P. V., et al., "Reliability of non-destructive test techniques in the inspection of pipelines used in the oil industry", *International Journal of Pressure Vessels and Piping*, v. 85, n. 11, pp. 745 – 751, 2008.

[3] FENG, Q., LI, R., NIE, B., et al., "Literature Review: Theory and Application of In-Line Inspection Technologies for Oil and Gas Pipeline Girth Weld Defection", *Sensors*, v. 17, n. 1, 2017.

[4] KESHARAJU, M., NAGARAJAH, R., "Feature selection for neural network based defect classification of ceramic components using high frequency ultrasound", *Ultrasonics*, v. 62, pp. 271 – 277, 2015.

[5] HER, S.-C., LIN, S.-T., "Non-Destructive Evaluation of Depth of Surface Cracks Using Ultrasonic Frequency Analysis", *Sensors*, v. 14, n. 9, pp. 17146–17158, 2014.

[6] SU, L., ZHA, Z., LU, X., et al., "Using BP network for ultrasonic inspection of flip chip solder joints", *Mechanical Systems and Signal Processing*, v. 34, n. 1, pp. 183 – 190, 2013.

[7] SHULL, P. J., *Nondestructive Evaluation Theory, Techniques and Applications*. Marcel Dekker, Inc.: The Pennsylvania State University, Altoona, Pennsylvania, 2001.

[8] FELICE, M. V., FAN, Z., "Sizing of flaws using ultrasonic bulk wave testing: A review", *Ultrasonics*, v. 88, pp. 26 – 42, 2018.

[9] CHI, D., GANG, T., "Shallow Buried Defect Testing Method Based on Ultrasonic TOFD", *Journal of Nondestructive Evaluation*, v. 32, n. 2, pp. 164–171, Jun 2013.

[10] RUSSEL, S. J., NOVIG, P., *Artificial Intelligence: A Modern Approach*. 3rd ed. Pearson: Upper Saddle River, NJ, USA, 2013.

[11] COCKBURN, I. M., HENDERSON, R., STERN, S., *The Impact of Artificial Intelligence on Innovation*, Working Paper 24449, National Bureau of Economic Research, March 2018.

[12] HAYKIN, S., *Neural Networks and Learning Machines*. 3rd ed. Prentice Hall: McMaster University, Canada, 2008.

[13] POULARIKAS, A., DORF, R., GRIGORIAN, A., *Transforms and Applications Handbook*. CRC Press: Boca Raton, USA, 2010.

[14] HYVÄRINEN, A., OJA, E., "Independent component analysis: algorithms and applications", *Neural Networks*, v. 13, n. 4, pp. 411 – 430, 2000.

[15] SIMAS FILHO, E. F., ALBUQUERQUE, M. C. S., FARIAS, C. T. T., "Sistema Neural de Apoio à Decisão na Inspeção por Ultrassom Utilizando os Componentes Principais das Transformadas de Fourier, Cossenos e Wavelet". In: *Anais do XIX Congresso Brasileiro de Automática, CBA 2012*, Campina Grande, PB, Brazil, 2012.

[16] DA C. CRUZ, F., D'ALMEIDA, M. S., SIMAS FILHO, E. F., et al., "Sistema de Classificação Baseado em Máquina de Vetor de Suporte para Auxílio ao Diagnóstico em Inspeções por Ultrassom Utilizando Ondas Guiadas". In: *13ª Conferência sobre Tecnologia de Equipamentos*, Cabo de Santo Agostinho, PE, Brazil, 2015.

[17] BARRY, T., KESHARAJU, M., NAGARAJAH, C., et al., "Defect characterisation in laminar composite structures using ultrasonic techniques and artificial neural networks", *Journal of Composite Materials*, v. 50, n. 7, pp. 861–871, 2016.

[18] The American Society for Nondestructive Testing, *Recommended Practice No. SNT-TC-1A: Personnel Qualification and Certification in Nondestructive Testing (2016)*.

[19] Associação Brasileira de Ensaios Não Destrutivos e Inspeção, *Qualificação e certificação de pessoas em ensaios não destrutivos NA-001*.

[20] Associação Brasileira de Normas Técnicas, *Non-destructive testing - Qualification and certification of NDT personnel (ISO 9712:2012, IDT)*.

[21] MESSLER JR., R. W., *Principles of Welding*. WILEY-VCH Verlag GmbH & Co. KGaA: Weinheim, Germany, 2004.

[22] KOU, S., *Welding Metallurgy*. 2nd ed. John Wiley & Sons: Hoboken, NJ, USA, 2002.

[23] BLITZ, J., SIMPSON, G., *Ultrasonic Methods for Non-destructive Testing*. Chapman & Hall: 2-6 Boundary Row, London SE1 8HN, UK, 1996.

[24] SANTIN, J. L., *Ultra-som: Técnica e Aplicação*. Optagraf Gráfica e Editora: Curitiba, Brazil, 2003.

[25] CHARLESWORTH, J. P., TEMPLE, J. A. G., *Engineering Applications of Ultrasonic Time-of-Flight Diffraction*. Research Studies Press LTD.: Baldock, Hertfordshire, England, 2001.

[26] VEIGA, J. L. B. C., CARVALHO, A. A. D., SILVA, I. C. D., et al., "The use of artificial neural network in the classification of pulse-echo and TOFD ultra-sonic signals", *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, v. 27, pp. 394 – 398, 12 2005.

[27] SEYEDTABAII, S., "Performance Evaluation of Neural Network Based Pulse-Echo Weld Defect Classifiers", *Measurement Science Review*, v. 12, n. 5, pp. 168 – 174, 2012.

[28] LIU, J., XU, G., REN, L., et al., "Defect intelligent identification in resistance spot welding ultrasonic detection based on wavelet packet and neural network", *The International Journal of Advanced Manufacturing Technology*, v. 90, n. 9, pp. 2581–2588, Jun 2017.

[29] SIMAS FILHO, E. F., SOUZA, Y. N., LOPES, J. L., et al., "Decision support system for ultrasound inspection of fiber metal laminates using statistical signal processing and neural networks", *Ultrasonics*, v. 53, n. 6, pp. 1104 – 1111, 2013.

[30] MURTA, R. H. F., VIEIRA, F. D. A., SANTOS, V. O., et al., "Welding Defect Classification from Simulated Ultrasonic Signals", *Journal of Nondestructive Evaluation*, v. 37, n. 3, pp. 40, May 2018.

[31] HABIBPOUR-LEDARI, A., HONARVAR, F., "Three Dimensional Characterization of Defects by Ultrasonic Time-of-Flight Diffraction (ToFD) Technique", *Journal of Nondestructive Evaluation*, v. 37, n. 1, pp. 14, Feb 2018.

[32] DOUGHERTY, G., *Pattern Recognition and Classification: An Introduction*. Springer Publishing Company, Incorporated, 2013.

[33] POWERS, D. M. W., *Evaluation: From Precision, Recall and F-Factor to ROC, Informedness, Markedness & Correlation*, Tech. rep., School of Informatics and Engineering Flinders, University of South Australia, 2011.

[34] LABATUT, V., CHERIFI, H., "Accuracy Measures for the Comparison of Classifiers". In: *The 5th Conference on Information Technology*, Amman, Jordan, May 2011.

[35] LYONS, R. G., *Understanding Digital Signal Processing (2nd Edition)*. Prentice Hall PTR: Upper Saddle River, NJ, USA, 2004.

[36] CHICCO, D., "Ten quick tips for machine learning in computational biology", *BioData Mining*, v. 10, n. 1, pp. 35, Dec 2017.

[37] LI, S., WANG, Z., WANG, J., "Study of Face Orientation Recognition Based on Neural Network", *International Journal of Pattern Recognition and Artificial Intelligence*, v. 32, n. 11, pp. 1856015, 2018.

[38] CHATTOPADHYAY, S., CHATTOPADHYAY, G., "Conjugate gradient descent learned ANN for Indian summer monsoon rainfall and efficiency

assessment through Shannon-Fano coding", *Journal of Atmospheric and Solar-Terrestrial Physics*, v. 179, pp. 202 – 205, 2018.

[39] PIEDAD, E. J., LARADA, J. I., POJAS, G. J., et al., "Postharvest classification of banana (Musa acuminata) using tier-based machine learning", *Postharvest Biology and Technology*, v. 145, pp. 93 – 100, 2018.

[40] NADERPOUR, H., POURSAEIDI, O., AHMADI, M., "Shear resistance prediction of concrete beams reinforced by FRP bars using artificial neural networks", *Measurement*, v. 126, pp. 299 – 308, 2018.

[41] LIM, H. S., KANG, Y. T., "Estimation of finish cooling temperature by artificial neural networks of backpropagation during accelerated control cooling process", *International Journal of Heat and Mass Transfer*, v. 126, pp. 579 – 588, 2018.

[42] SINGH, A. K., KUMAR, B., SINGH, S. K., et al., "Multiple watermarking technique for securing online social network contents using Back Propagation Neural Network", *Future Generation Computer Systems*, v. 86, pp. 926 – 939, 2018.

[43] KINOSHITA, K., OHNO, S., WAKITANI, S., "Design of neural network PID controller based on E-FRIT". In: *2017 56th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE)*, pp. 1183–1186, Sept 2017.

[44] LEVENBERG, K., "A method for the solution of certain non-linear problems in least squares", *Quaterly of Applied Mathematics*, v. 2, n. 2, pp. 164 – 168, 1944.

[45] MARQUARDT, D., "An Algorithm for Least-Squares Estimation of Nonlinear Parameters", *Journal of the Society for Industrial and Applied Mathematics*, v. 11, n. 2, pp. 431–441, 1963.

[46] GAVIN, H. P., *The Levenberg-Marquardt method for nonlinear least squares curve-fitting problems*, Tech. rep., Duke University, 2017, `http://people.duke.edu/~hpgavin/ce281/lm.pdf`.

[47] YU, H., WILAMOWSKI, B., "Levenberg-Marquardt Training", In: *Intelligent Systems*, 2nd ed., chap. 12, pp. 12–1 to 12, CRC Press, 01 2011.

[48] RIEDMILLER, M., BRAUN, H., "RPROP - A Fast Adaptive Learning Algorithm". In: *Proc. of the Int. Symposium on Computer and Information Science*, v. VII, 12 1992.

[49] HUANG, G.-B., ZHU, Q.-Y., SIEW, C.-K., "Extreme learning machine: Theory and applications", *Neurocomputing*, v. 70, n. 1, pp. 489 – 501, 2006.

[50] CAO, W., WANG, X., MING, Z., et al., "A review on neural networks with random weights", *Neurocomputing*, v. 275, pp. 278 – 287, 2018.

[51] HUANG, G., ZHOU, H., DING, X., et al., "Extreme Learning Machine for Regression and Multiclass Classification", *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, v. 42, n. 2, pp. 513–529, 2012.

[52] AGGARWAL, C. C., *Neural Networks and Deep Learning*. Springer International Publishing: Cham, Switzerland, 2018.

[53] HUANG, G.-B., "An Insight into Extreme Learning Machines: Random Neurons, Random Features and Kernels", *Cognitive Computation*, v. 6, n. 3, pp. 376–390, Sep 2014.

[54] ROGBEN, L., *Handbook of linear algebra*. Chapman & Hall/CRC: Boca Raton, FL, USA, 2007.

[55] HUANG, G.-B., WANG, D. H., LAN, Y., "Extreme learning machines: a survey", *International Journal of Machine Learning and Cybernetics*, v. 2, n. 2, pp. 107–122, Jun 2011.

[56] TAO, X., ZHOU, X., HE, Y. L., et al., "Impact of Variances of Random Weights and Biases on Extreme Learning Machine", *Journal of Software*, v. 11, pp. 440 – 454, 2016.

[57] PRASAD, R., DEO, R. C., LI, Y., et al., "Soil moisture forecasting by a hybrid machine learning technique: ELM integrated with ensemble empirical mode decomposition", *Geoderma*, v. 330, pp. 136 – 161, 2018.

[58] HONG, Y., CHEN, S., ZHANG, Y., et al., "Rapid identification of soil organic matter level via visible and near-infrared spectroscopy: Effects of two-dimensional correlation coefficient and extreme learning machine", *Science of The Total Environment*, v. 644, pp. 1232 – 1243, 2018.

[59] ZHU, Z., ZHU, X., KONG, F., et al., "A rapid method on identifying disqualified raw goat's milk based on total bacterial count by using dielectric spectra", *Journal of Food Engineering*, v. 239, pp. 40 – 51, 2018.

[60] SHARMA, S., GHANEKAR, U., "A hybrid technique to discriminate Natural Images, Computer Generated Graphics Images, Spliced, Copy Move tampered images and Authentic images by using features and ELM classifier", *Optik*, v. 172, pp. 470 – 483, 2018.

[61] ZONG, W., HUANG, G.-B., "Face recognition based on extreme learning machine", *Neurocomputing*, v. 74, n. 16, pp. 2541 – 2551, 2011, Advances in Extreme Learning Machine: Theory and Applications Biological Inspired Systems. Computational and Ambient Intelligence.

[62] YANG, XIQI, ZHANG, QINGFENG, YANG, XIYU, et al., "Edge detection in Cassini astronomy image using Extreme Learning Machine", *MATEC Web Conf.*, v. 189, pp. 06007, 2018.

[63] HUANG, G., HUANG, G.-B., SONG, S., et al., "Trends in extreme learning machines: A review", *Neural Networks*, v. 61, pp. 32 – 48, 2015.

[64] KASUN, L., ZHOU, H., HUANG, G.-B., et al., "Representational Learning with ELMs for Big Data", *IEEE Intelligent Systems*, v. 28, pp. 31–34, 11 2013.

[65] FERREIRA, J., FERRO, M., FERNANDES, B., et al., "Extreme learning machine autoencoder for data augmentation". In: *2017 IEEE Latin American Conference on Computational Intelligence (LA-CCI)*, pp. 1–6, Nov 2017.

[66] DING, S., ZHANG, N., ZHANG, J., et al., "Unsupervised extreme learning machine with representational features", *International Journal of Machine Learning and Cybernetics*, v. 8, 03 2015.

[67] SUN, K., ZHANG, J., ZHANG, C., et al., "Generalized extreme learning machine autoencoder and a new deep neural network", *Neurocomputing*, v. 230, pp. 374 – 381, 2017.

[68] KASUN, L. L. C., YANG, Y., HUANG, G., et al., "Dimension Reduction With Extreme Learning Machine", *IEEE Transactions on Image Processing*, v. 25, n. 8, pp. 3906–3918, Aug 2016.

[69] IBRAHIM, D., "Chapter 1 - Introduction", In: *ARM-Based microcontroller projects using MBED*, chap. 1, pp. 1 – 7, Newnes, 2019.

[70] TORRES, P. F., COSTA, A. F. P., CHAAR JUNIOR, V. L., et al., "A mobile educational tool designed for teaching and dissemination of grid connected photovoltaic systems", *Computers & Electrical Engineering*, v. 76, pp. 168 – 182, 2019.

[71] BARKUNAN, S., BHANUMATHI, V., SETHURAM, J., "Smart sensor for automatic drip irrigation system for paddy cultivation", *Computers & Electrical Engineering*, v. 73, pp. 180 – 193, 2019.

[72] KOS, M., ROJC, M., ŽGANK, A., et al., "A speech-based distributed architecture platform for an intelligent ambience", *Computers & Electrical Engineering*, v. 71, pp. 818 – 832, 2018.

[73] AL-KOFAHI, M. M., AL-SHORMAN, M. Y., AL-KOFAHI, O. M., "Toward energy efficient microcontrollers and Internet-of-Things systems", *Computers & Electrical Engineering*, v. 79, pp. 106457, 2019.

[74] GRIDLING, G., WEISS, B., "Introduction to Microcontrollers", Vienna University of Technology.

[75] RAJ, S., CHAND, G. P., RAY, K. C., "ARM-based arrhythmia beat monitoring system", *Microprocessors and Microsystems*, v. 39, n. 7, pp. 504 – 511, 2015.

[76] GARCIA-BREIJO, E., ATKINSON, J., GIL-SANCHEZ, L., et al., "A comparison study of pattern recognition algorithms implemented on a microcontroller for use in an electronic tongue for monitoring drinking waters", *Sensors and Actuators A: Physical*, v. 172, n. 2, pp. 570 – 582, 2011.

[77] NEZ CIVERA, J. I., BREIJO, E. G., MIRÓ, N. L., et al., "Artificial neural network onto eight bit microcontroller for Secchi depth calculation", *Sensors and Actuators B: Chemical*, v. 156, n. 1, pp. 132 – 139, 2011.

[78] BAYINDIR, R., SAGIROGLU, S., COLAK, I., "An intelligent power factor corrector for power system using artificial neural networks", *Electric Power Systems Research*, v. 79, n. 1, pp. 152 – 160, 2009.

[79] CHO, D., HAM, J., OH, J., et al., "Detection of Stress Levels from Biosignals Measured in Virtual Reality Environments Using a Kernel-Based Extreme Learning Machine", *Sensors*, v. 17, n. 10, 2017.

[80] UÇAR, A., OZALP, R., "Efficient android electronic nose design for recognition and perception of fruit odors using Kernel Extreme Learning Machines", *Chemometrics and Intelligent Laboratory Systems*, v. 166, pp. 69 – 80, 2017.

[81] SUDA, N., LOH, D., *Machine Learning on ARM Cortex-M Microcontrollers*, Tech. rep., ARM, 2019.

[82] SAAD SAOUD, L., KHELLAF, A., "A neural network based on an inexpensive eight-bit microcontroller", *Neural Computing and Applications*, v. 20, n. 3, pp. 329–334, Apr 2011.

[83] Swift Navigation, *Piksi Datasheet*, march 2016.

[84] Robotis, *OpenCM 9.04 e-manual*, Available at `https://emanual.robotis.com/docs/en/parts/controller/opencm904/`.

[85] WEBSTER, J. G., NIMUNKAR, A. J., *Medical Instrumentation: Application and Design*. John Wiley & Sons: New Jersey, USA, 2020.

[86] CRUZ, F., SIMAS FILHO, E., ALBUQUERQUE, M., et al., "Efficient feature selection for neural network based detection of flaws in steel welded joints using ultrasound testing", *Ultrasonics*, v. 73, pp. 1 – 8, 2017.

[87] MANJULA, K., VIJAYAREKHA, K., VENKATRAMAN, B., "Quality Enhancement of Ultrasonic TOFD Signals from Carbon Steel Weld Pad with Notches", *Ultrasonics*, v. 84, pp. 264 – 271, 2018.

[88] VAN HEESWIJK, M., MICHE, Y., "Binary/ternary extreme learning machines", *Neurocomputing*, v. 149, pp. 187 – 197, 2015, Advances in neural networks Advances in Extreme Learning Machines.

[89] SILVA, L. C., OLIVEIRA, M., SIMAS FILHO, E. F., et al., "Classificação de falhas em solda de chapas de aço por meio de máquina de aprendizado extremo aplicada a inspeções não-destrutivas por ultrassom". In: *14ª Conferência sobre Tecnologia de Equipamentos*, Rio de Janeiro, RJ, Brazil, 2017.

[90] SILVA, L. C., SIMAS FILHO, E. F., ALBUQUERQUE, M. C., et al., "Segmented analysis of time-of-flight diffraction ultrasound for flaw detection in welded steel plates using extreme learning machines", *Ultrasonics*, v. 102, pp. 106057, 2020.

[91] GALASSI, M., DAVIES, J., THEILER, J., et al., *GNU Scientific Library Reference Manual*, 3rd ed., 2009.

[92] *Basic Linear Algebra Subroutines Technical (BLAST) Forum Standard*, Tech. rep., University of Tennessee, 2001.

[93] STALLMAN, R. M., *Using the GNU Compiler Collection*. GNU Press: Boston, USA, 2019.

Appendix A

ELM Implementation

This appendix presents details on the ELM implementation regarding the library and its respective API developed for training ANNs in microcontrollers using the Extreme Learning Machine approach. This library relies on

- The popularity of 32-bit architecture processors capable of hardware multiplications;
- The mentioned advantages of the ELM theory for an embedded platform

The chosen microcontrolled platform for this work is the family STM32F4 from ST Microelectronics. STM32s are abundant in market for 32-bit microcontrollers, and offer several models with different packages and configurations, all of them with ARM cores at their CPUs. The family STM32F4 has arithmetic and logic units (ALUs) with instructions for floating point operations executed in dedicated hardware. Evaluation boards are also available to shorten development time.

The API for microcontrollers was built based on GNU Scientific Library (GSL) [91]. GSL consists in a collection of libraries built to support numerically scientific development in C and C++ languages. Also, it has support for BLAS (Basic Linear Algebra Subprograms), consisting in routines used in vector and matrix operations commonly demanded in linear algebra, like dot products, matrix multiplications, scaling, etc [92]. The support for vector and matrix operations along with libraries for linear algebra are relevant for the current problem, because the training step in ELM mainly consists of the computation of Moore-Penrose pseudoinverse which in its turn consists of matrix multiplication. Whenever possible, matrices and vectors

were stored in 32-bit flot single precision format in order to leverage the hardware capabilities of the microcontroller. To the date of this work's conclusion, ARM's native CMSIS library for dealing with matrices lack inplace operations, so that GSL operations seemed better suited for the work.

In order to have and ELM-based classifier running in a microcontroller, an API was coded in C++. Conceiving the routines took into account two main factors:

1. The API should provide means to train an ANN following the Extreme Learning Machine paradigm, building a network with one hidden layer and random weights for the connections between input and hidden layer, and store the trained network in memory. The user may inform the dimensions n and m of the input and output data respectively as well as the hidden neuron count.

2. The API must provide means to access the trained network to make it available in firmware as new data is presented to network.

Following these requirements, the library comprises three classes:

1. **Elm**. A class whose object allocates memory for random weights, random bias and output weights and contains the methods for training and using the network.

2. **Slfn**. A struct with general parameters for setting a network, such as node and neuron count and neuron type (additive or nonlinear limiter).

3. **Organizer**. An auxiliary class used for collecting samples in training set, targets and test sample.

Figures A.1, A.2 and A.3 depicts class diagram for the aforementioned classes in Unified Modelling Language (UML) representation of the Elm class implemented in C++. The first row in each diagram is the name of the class, the second row contains its attributes and finally the third row shows the class' methods). Attributes and methods have their types displayed as well. The preceding signs give access information. The "+" sign indicates public attribute or methods, whereas the "-" sign is used for private ones.

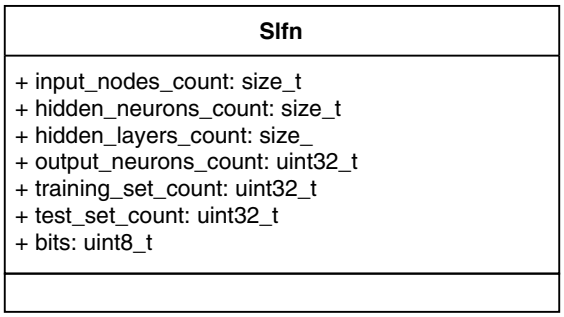


Figure A.1: UML class diagram for Slfn class.

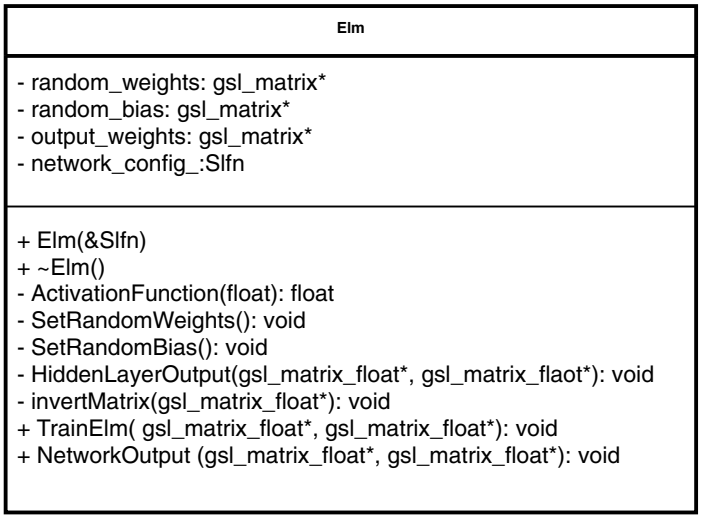


Figure A.2: UML class diagram for Elm class.

The GSL distribution is ready to be compiled and installed on Linux x86 environments. In order to build GSL libraries to run with the ELM API for microcontrollers, GSL must be cross-compiled to another platform, so that functions based on the operational system can now run in an embedded-application binary interface (EABI).

Once determined the target family, the GSL files were cross-compiled to the ARM platform using the GNU Compiler Collection (GCC) [93]. The parameters to pass to the compiler linker are the flag for suppressing system calls and the option to use hardware in the ARM core for floating point calculations.

The GSL library is provided with the library as a compiled binary library, instead of source files. This avoids recompilation of the GSL library every time a new compile operation is done by the user while working with ELM API, since the GLS library is not meant to be altered.

Dynamic allocation and deallocation of memory blocks was strategically used as

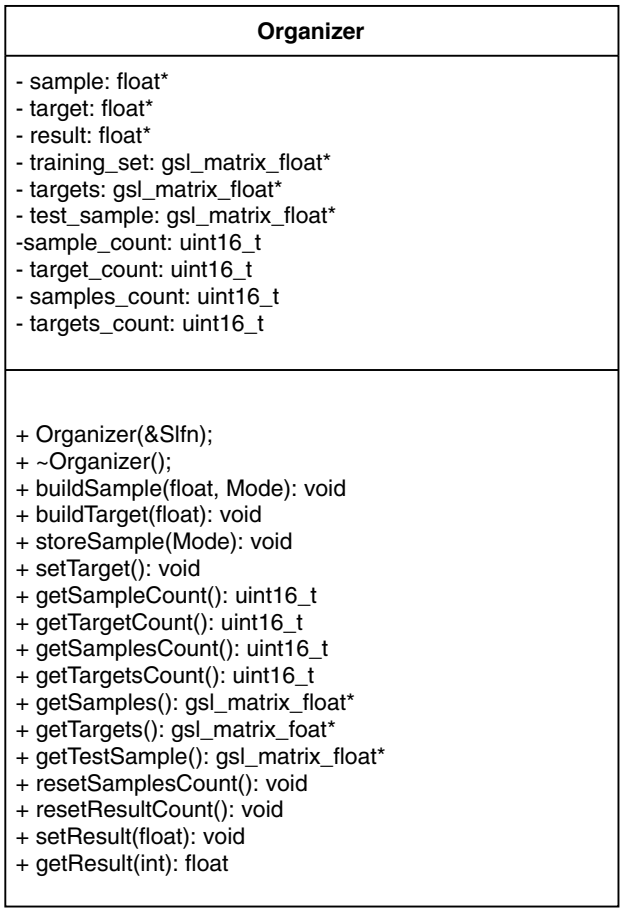


Figure A.3: UML class diagram for Organizer class.

a way to best use the available memory, meaning that:

- Some variables are not immediatly assigned to a memory block. It is the case of "output_weights" in Elm class (Figure A.2 which is not assigned to any memory block upon a call to the class constructor. This happens only right before its calculation at the training step;
- Variables are deallocated as soon as their content is no longer needed, thus there is no wait until the destruction of the object in which they belong.

Since this operation uses the region known as heap (data memory used for dynamic allocation), the data memory usage information at compile time (corresponding to the stack, where declared variables are stored) doesn't reflect the real memory usage. Table A.1 shows the memory demanded, in bytes, by the variables that deal with large amounts of data in this implementation (T in this Table stands for the test set size). Most variables are 32-bit float types, thus the multiplication by 4 (by

8 when in double format). When the training set is much larger then the hidden neurons count (Equation 3.29), there is less demand for heap allocation (although a trade-off concerning memory occupation by a huge training set shall be considered).

Organizer		
training_set	$4Nn$	
targets	$4Nm$	
test_sample	$4n$	
result	$4Tm$	
Elm		
random_weights	$4Ln$	
random_bias	$4L$	
output_weights	$4Lm$	
Training step		
hidden_layer_outputs	$4NL$	
regulation matrix	$4L^2$	$N > 20L$
	$4N^2$	$N < 20L$
auxiliary regulation matrix	0	$N > 20L$
	$4Nm$	$N < 20L$
matrix for inversion (must be in double format)	$8L^2$	$N > 20L$
	$8N^2$	$N < 20L$
inverted_matrix (must be in double format)	$8L^2$	$N > 20L$
	$8N^2$	$N < 20L$
Test step		
hidden_layer_output	$4L$	

Table A.1: Memory usage in ELM implementation.

A.1 Validation

The tests with the library were executed using simulation-generated data and real data provided by a device connected to the platform. The purpose was to test the developed classes and inheritance of GSL libraries in the project.

A.1.1 Simulation and performance assessment

A series of performance assessment trials were run in the microcontroller in order to verify the correct execution of the library components. An implementation in MATLAB environment was used as ground truth, so that the same training and test sets were presented to both the MATLAB and microcontroller networks. Both networks had the same random weights assigned to the same hidden neuron count. Another subject of interest in the tests was how poor the microcontroller network would perform regarding using 32-bit float format when compared to 64-bit double format used in MATLAB.

The data set was generated at MATLAB and transmitted to the microcontroller by means of serial protocol. Performance was measured in terms of root mean squared error (RMSE) for each output neuron, calculated on the test set. The trials are described in Table A.2, which contains the training set size, test set size, hidden neurons count and output neurons count for each test. A picture of each test case is also present for illustration purpose.

The RMSE values shown in each trial indicate that the library works properly when compared to the implementation on a personal computer.

A.1.2 Tests with real data

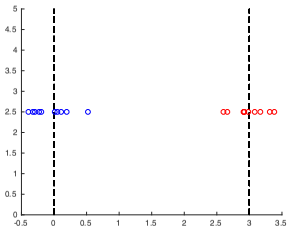
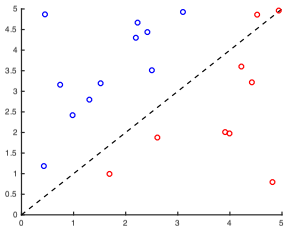
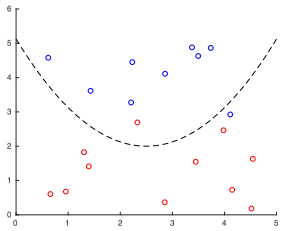
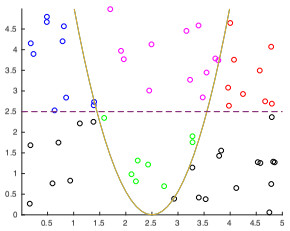
An experiment with the features of the STM32F4 Discovery was carried out. The goal was to identify the inclination of the board with respect to an axis, using its accelerometer. The ELM network used the accelerometer data as input, and provided an output which indicated what as the current inclination of the board with respect to roll angle (see Figure A.4).

The system setup was configured as follows:

- **Input layer**: the 3-axis accelerometer measures static acceleration, which only considers the effect of gravity. This way, the gravity vector is decomposed into 3 cartesian coordinates, forming a sample with dimension 3. Each dimension was represented by an input node, resulting in a 3-node input layer.

- **Training set**: the training set was built with 40 samples. For each sample, a respective target value was assigned. The values were associated to 2 different

Table A.2: Performance trials.

	1-D data, linearly separable	**2-D data, linearly separable**	**2-D data, non-linearly separable**	**2-D data, multiclass**
training set	40	40	40	60
test set	20	20	20	30
hidden neurons	10	10	10	20
output neurons	2	2	2	5
RMSE	$[2.8892.868] \times 1e^{-7}$	$[0.3380.433] \times 1e^{-7}$	$[0.2330.473] \times 1e^{-7}$	$[0.1530.2210.2810.6010.233] \times 1e{-}7$
Problem illustration				

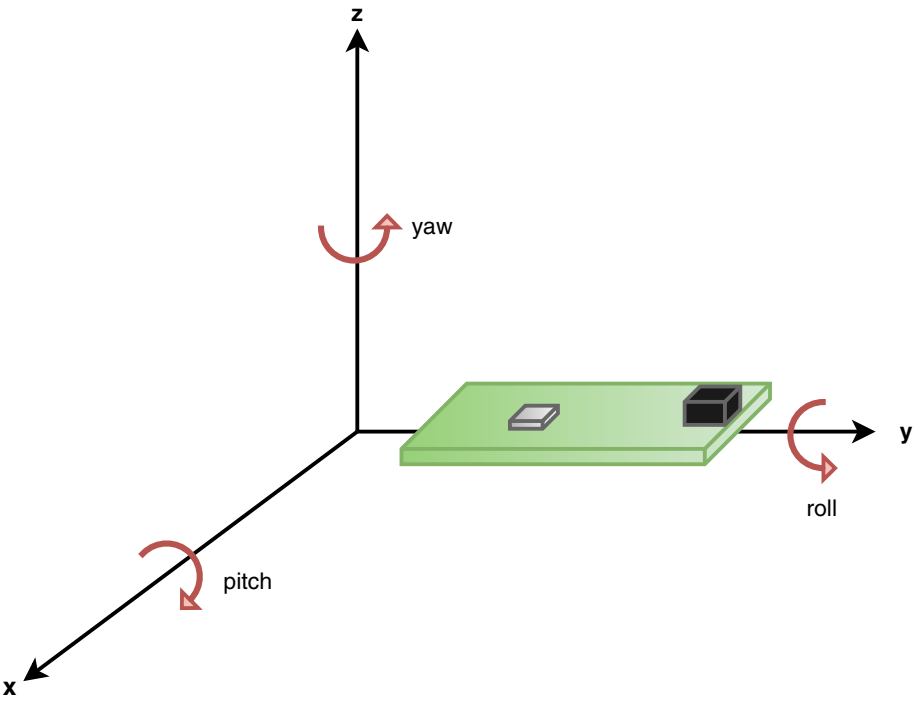


Figure A.4: Reference frame for the board test.

I/Os, each of which connected to a different LED. The samples were taken at different angles with respect to the horizontal plan, and the reference targets were positive inclination and negative inclination.

- **Hidden layer**: 10 neurons at the hidden layer.
- **Output layer**: 2 neurons at the output layer. Each neuron was responsible for a inclination information with respect to the horizontal plane.
- **Test set**: 20 samples were acquired for the test set.

This preliminary evaluation resulted in accuracy of 90%, with an error detected for each class.

The library built for this work is an open source project. It can be download, forked and shared by the scientific and engineering community at `https://github.com/lucascsilva/elm_mcu`.

Appendix B

Published and submitted papers

Following are the published papers during this Thesis' research period.

- OLIVEIRA, M., SILVA, L. C., SIMAS FILHO, E. F., et al., Detecção de "Novidades" na Classificação de Defeitos em Junta Soldada de de Aço SAE1020. In: *XXXIV Congresso Nacional de Ensaios Não Destrutivos e Inspeção*, São Paulo, SP, Brazil, 2016.

  Abstract:

  A classificação automática de defeitos em juntas soldadas utilizando inspeção ultrassônica vem sendo estudada há alguns anos. Diversas técnicas como redes neurais artificiais, árvores de decisão e discriminantes lineares foram utilizadas com sucesso. Entretanto, normalmente, os sistemas de classificação são projetados para identificar sinais que pertencem a um número limitado de classes, estabelecido previamente a partir das amostras disponíveis para treinamento. Se na operação do sistema ocorrer uma condição diferente das existentes no treinamento (classe de sinal desconhecida), ela será associada à condição conhecida mais semelhante. Essa limitação pode representar um problema quando o conjunto de treinamento não é representativo de todas as possíveis condições do material. Este trabalho propõe utilizar uma técnica de projeto de sistemas de classificação denominada "detecção de novidades" para identificação de defeitos em juntas soldadas a partir de inspeções por ultrassom pela técnica *TOFD*. Neste caso, o classificador será capaz de reconhecer uma amostra de entrada que não pertence a nenhum dos padrões

de treinamento, possibilitando maior flexibilidade ao sistema. Sinais de ultrassom *TOFD*, tomados por varredura unma chapa de aço SAE1020 soldada onde existem diferentes tipos de defeito, serão utilizados para alimentar um sistema de classificação por redes neurais artificiais.

- SILVA, L. C., OLIVEIRA, M., SIMAS FILHO, E. F., et al., Classificação de Falhas em Solda de Chapas de Aço por meio de Máquina de Aprendizado Extremo Aplicada a Inspeções Não-Destrutivas por Ultrassom. In: *14ª Conferência sobre Tecnologia de Equipamentos*, Rio de Janeiro, RJ, Brazil, 2017.

  Abstract:

  É utilizado um método de treinamento de Redes Neurais Artificiais conhecido como Máquinas de Aprendizado Extremo para classificação de defeitos em inspeções não-destrutivas por ultrassom em regiões de solda de chapas de aço-carbono. Essa estratégia de treinamento visa a diminuição do esforço computacional necessário para o desenvolvimento do classificador, sendo adequada para o uso em sistemas embarcados dedicados. As assinaturas espectrais (após transformada de Fourier) dos sinais de defeitos comumente observados em soldas tais como falta de fusão, porosidade e inclusão de escória, são utilizadas para projetar e testar o método de classificação proposto. O método é comparado com técnicas de classificação baseadas em topologias de Redes Neurais Artificiais estabelecidas na literatura. Os resultados obtidos indicam que o uso de Máquinas de Aprendizado Extremo é vantajoso sob o ponto de vista do tempo de treinamento do classificador em relação aos outros algoritmos. Deste modo, o método proposto se apresenta como uma boa opção para treinamento em campo utilizando dispositivos portáteis dedicados.

- L. C. Silva, E. F. Simas Filho, M. C. Albuquerque, I. C. Silva, C. T. Farias, Segmented analysis of time-of-flight diffraction ultrasound for flaw detection in welded steel plates using extreme learning machines, *Ultrasonics* 102 (2020) 106057. doi:https://doi.org/10.1016/j.ultras.2019.106057.

  Abstract:

  This work investigates the application of extreme learning machine, a fast training neural network model, for an ultrasound nondestructive evaluation

decision support system. A novel segmented analysis of time-of-flight diffraction ultrasound signals is proposed in order to produce high flaw detection efficiency and low computational requirements, making it possible to be used in embedded applications. The frequency contents of TOFD signals temporal segments, estimated using the discrete Fourier transform, were used to feed the classification system. The test objects consisted of a set of SAE 1020 welded carbon steel plates, in which occur four types of defects. The obtained experimental results indicate that the proposed method is able to combine high accuracy,fast training and full exploration of the TOFD signal information.

Next is a submitted journal paper under revision:

- L. C. Silva, E. F. Simas Filho, M. C. Albuquerque, I. C. Silva, C. T. Farias, Embedded Decision Support System for Ultrasound Nondestructive Evaluation based on Extreme Learning Machines.

  Abstract:

  Decision support systems are an important asset in nondestructive evaluation, as they enhance an inspector's assessment of an equipment. Training such systems for specific characteristics derived from production processes, such as defects, is often run on a personal computer environment, which may slow the assessment task. This work investigates the feasibility of embedding extreme learning machines in microcontrollers, in order to have a decision support system capable of online training for an ultrasound nondestructive evaluation. Principal component analysis and autoencoders were evaluated as dimensionality reduction methods in a data set acquired from welded SAE 1020 carbon steel plates containing four types of defects. In order to optimize memory usage, a binary-encoding random weight approach is proposed, whose results were tested over four different probability density functions used to generate the random weights. The obtained experimental results matched the accuracy of a previous work, run in computer environment for higher dimensionality data. In this work a dimensionality reduction of 75% and elapsed time for system training of less than 0.5 ms in microcontroller were achieved, indicating the suitability of such embedded networks for the studied case.