Pontifícia Universidade Católica
do Rio de Janeiro

Carolina de Oliveira Contente

Vibration monitoring of mechanical systems using deep and shallow learning on edge-computers

Dissertação de Mestrado

Dissertation presented to the Programa de Pós-graduação em Engenharia Mecânica of PUC-Rio in partial fulfillment of the requirements for the degree of Mestre em Engenharia Mecânica.

Advisor: Prof. Helon Vicente Hultmann Ayala

Rio de Janeiro
May 2022

Pontifícia Universidade Católica
do Rio de Janeiro

Carolina de Oliveira Contente

Vibration monitoring of mechanical systems using deep and shallow learning on edge-computers

Dissertation presented to the Programa de Pós-graduação em Engenharia Mecânica of PUC-Rio in partial fulfillment of the requirements for the degree of Mestre em Engenharia Mecânica. Approved by the Examination Committee:

Prof. Helon Vicente Hultmann Ayala
Advisor
Departamento de Engenharia Mecânica – PUC-Rio

Prof. Arthur Martins Barbosa Braga
Departamento de Engenharia Mecânica – PUC-Rio

Prof. Roberto Zanetti Freire
PUC-PR

Rio de Janeiro, May the 12th, 2022

Carolina de Oliveira Contente

The author majored in Mechanical Engineering by the Universidade Federal do Pará (Belém, PA).

Bibliographic data

Contente, Carolina de Oliveira

Vibration monitoring of mechanical systems using deep and shallow learning on edge-computers / Carolina de Oliveira Contente; advisor: Helon Vicente Hultmann Ayala. – 2022.

59 f: il. color. ; 30 cm

Dissertação (mestrado) - Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Engenharia Mecânica, 2022.

Inclui bibliografia

1. Engenharia Mecânica – Teses. 2. Monitoramento da Integridade Estrutural. 3. Identificação de Sistemas. 4. Aprendizado de Máquina. 5. Aprendizado Supervisionado. 6. Aprendizado não Supervisionado. I. Ayala, Helon Vicente Hultmann. II. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Engenharia Mecânica. III. Título.

CDD: 621

To my parents, for their support
and encouragement.

Acknowledgments

I would like to first thank my advisor, Helon Ayala, for his guidance and patience during this whole process. Special thanks to Professor Hans Weber, Professor Carlos Llanos and Marlon Sodre for their contribution in this dissertation. Also I would like to thank my good friend Ingrid Pires for her contribution and friendship. Thanks to all faculty, staff and friends from PUC-Rio.

Thanks to my parents, Wania and Francisco, and my brother Gustavo for all the support they gave me. Thanks for always trust in me and support me in my ventures. Everything I do and everything I accomplish is for them.

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

Abstract

Contente, Carolina de Oliveira; Ayala, Helon Vicente Hultmann (Advisor). **Vibration monitoring of mechanical systems using deep and shallow learning on edge-computers**. Rio de Janeiro, 2022. 59p. Dissertação de Mestrado – Departamento de Engenharia Mecânica, Pontifícia Universidade Católica do Rio de Janeiro.

Structural health monitoring has been the focus of recent developments in vibration-based assessment and, more recently, in the scope of the internet of things as measurement and computation become distributed. Data has become abundant even though the transmission is not always feasible, especially in remote applications. It is thus essential to devise data-driven model workflows that ensure the best compromise between model accuracy for condition assessment and the computational resources needed for embedded solutions. This topic has not been widely used in the context of vibration-based measurements. In this context, the present research proposes two approaches for two applications, a static and a rotating one. In case one, a modeling workflow capable of reducing the dimension of autoregressive model features using principal component analysis and classifying this data using some of the main machine learning techniques such as logistic regression, support vector machines, decision tree classifier, k-nearest neighborhood and random forest classifier was proposed. The three-story building example was used to demonstrate the method's effectiveness, together with ways to assess the best compromise between accuracy and model size. In case two, a test rig composed of rotating inertias and slender connecting rods is used, and the monitoring solution was tested in an embedded GPU-based platform. The models implemented to effectively distinguish between different friction states were principal component analysis, deep autoencoder and artificial neural networks. Shallow models perform better concerning running time and accuracy in detecting faulty conditions.

Keywords

Structural Health Monitoring; Systems Identification; Machine Learning; Supervised Learning; Unsupervised Learning.

Resumo

Contente, Carolina de Oliveira; Ayala, Helon Vicente Hultmann. **Monitoramento de vibração em sistemas mecânicos usando aprendizado profundo e raso em computadores na ponta**. Rio de Janeiro, 2022. 59p. Dissertação de Mestrado – Departamento de Engenharia Mecânica, Pontifícia Universidade Católica do Rio de Janeiro.

O monitoramento de integridade estrutural tem sido o foco de desenvolvimentos recentes no campo da avaliação baseada em vibração e, mais recentemente, no escopo da internet das coisas à medida que medição e computação se tornam distribuídas. Os dados se tornaram abundantes, embora a transmissão nem sempre seja viável em frequências mais altas especialmente em aplicações remotas. Portanto, é importante conceber fluxos de trabalho de modelo orientados por dados que garantam a melhor relação entre a precisão do modelo para avaliação de condição e os recursos computacionais necessários para soluções incorporadas, tópico que não tem sido amplamente utilizado no contexto de medições baseadas em vibração. Neste contexto, a presente pesquisa propõe abordagens para duas aplicações: na primeira foi proposto um fluxo de trabalho de modelagem capaz de reduzir a dimensão dos parâmetros de modelos autorregressivos usando análise de componentes principais e classificar esses dados usando algumas técnicas de aprendizado de máquina como regressão logística, máquina de vetor de suporte, árvores de decisão, k-vizinhos próximos e floresta aleatória. O exemplo do prédio de três andares foi usado para demonstrar a eficácia do método. No segundo caso, é utilizado um equipamento de teste composto por inércias rotativas onde a solução de monitoramento foi testada em uma plataforma baseada em GPU embarcada. Os modelos implementados para distinguir eficazmente os diferentes estados de atrito foram análise de componentes principais, *deep autoencoders* e redes neurais artificiais. Modelos rasos têm melhor desempenho em tempo de execução e precisão na detecção de condições de falha.

Palavras-chave

Monitoramento da Integridade Estrutural; Identificação de Sistemas; Aprendizado de Máquina; Aprendizado Supervisionado; Aprendizado não Supervisionado.

Table of contents

List of figures

List of tables

List of Abreviations

ANN – Artificial Neural Network

AANN – Auto-associative Neural Network

AIC – Akaike Information Criterion

AR – Autoregressive Model

CNN – Convolutional Neural Netwrok

DAE – Deep Autoencoder

DTC – Decision Tree Classifier

Elu – Exponential Linear Unit

FA – Factor Analysis

FFANN – Feedforward Artificial Neural Network

FRF – Frequency Response Function

GG – Gini Gain

GKPCA – Greedy Kernel Principal Component Analysis

GPU – Graphics Processing Unit

IoT – Internet of Things

KNN – K-Nearest Neighborhood

KPCA – Kernel Principal Component Analysis

LGR – Logistic Regression

MSD – Mahalanobis Squared Distance

PAF – Partial Autocorrelation Function

PCA - Principal Component Analysis

RFC – Random Forest Classifier

RMSE – Root Mean Squared Error

RPM – Revolutions per Minute

SHM – Structural Health Monitoring

SLAM – Simultaneous Localization and Mapping

SPR – Statistical Pattern Recognition

SVD – Singular Value Decomposition

SVDD – Support Vector Data Description

SVM – Support Vector Machine

USA – United States of America

Long story short, I survived.

Taylor Swift, *Long Story Short.*

1
Introduction

1.1
Contextualization and motivation

Civil, nautical and aeronautical structures, among others, are subject to operational and environmental conditions that may change over time as material deterioration, wind and traffic loads, and earthquakes. These changes in operational and environmental conditions impose difficulties in the detection and identification of structural damage [1].

In this context, 7.5% of bridges in the United States are classified as structurally deficient, with many elements approaching their end of service life. And also, more than 30% of the 617,000 highway bridges need attention due to deteriorating conditions [2].

Several accidents were highlighted in the media, bringing attention to the importance of structural health monitoring (SHM) techniques and the need to monitor structures constantly. The I-35W St. Anthony Falls Bridge over the Mississippi River near downtown Minneapolis (Minnesota, USA) was one of these cases, illustrated in Figure 1.1. The bridge collapsed on August 1, 2007, bringing down 111 vehicles. Some studies were conducted years before the collapse, reporting damages to the bridge structure. But nothing that could prevent possible accidents was done.

The occurrence of failures in equipment and structures is usually associated with three leading causes: friction, impact, and fatigue. The monitoring of these structures is essential for the operation according to the project specifications, avoiding higher costs with parts replacement and interference in production progress.

It is fundamentally necessary to detect the failure to map the conditions of structural failures, which involves locating, quantifying, and estimating its magnitude [4]. The maintenance of structures must be performed regularly and can be divided into three stages: corrective, preventive, and predictive maintenance. Corrective maintenance aims to repair structures or equipment that have already suffered some type of damage. Preventive and predictive maintenance are similar but in predictive maintenance, no intrusion into the

Figure 1.1: Aerial view of the collapsed I-35W St. Anthony Falls Bridge over the Mississippi River, availabe in [3]

structure or equipment occurs, while preventive action only occurs when a possible failure is detected [5]. Predictive maintenance matters for this dissertation, where data is collected over time to monitor the structural condition to predict and prevent possible failures.

Inspection procedures are essential to provide assistance maintenance to structures that may be subjected to future failure. Brasiliano [6] stated that visual inspections are commonly made to adopted to assess structural conditions and detect damage. However, it has become inefficient and inadequate due to the high complexity of some structures, especially concerning identifying damages invisible to the human eye. Technologies have been developed to replace qualitative visual inspection and time-based maintenance procedures with more quantifiable and automated damage assessment processes, such as traditional vibration analysis, extensometry, ultrasound, eddy currents. In this context, SHM is considered, which aims to obtain information about the conditions of a given structure or parts of a structure.

According to Bornn et al. [7] and illustrated in Fig 1.2, there are four steps for SHM: (1) operational evaluation, (2) data acquisition, (3) feature extraction, and (4) statistical classification of the features; described by Figueiredo [8] as the statistical pattern recognition (SPR) paradigm.

First, an operational evaluation must be performed to determine the justification for monitoring the structure, how damage is defined for the monitored system, and determine the operational and environmental conditions under which the system functions. By doing this, some limitations are set on what

will be monitored and how this monitoring will be done, so the system aspects that feature the possible damage condition are well detected. The data acquisition part of the process consists in selecting the excitation methods such as sensor types, the number of sensors, places where the sensors will be fixed, and data acquisition hardware. The third step includes damage-sensitive features extraction. Extracting these features means fitting these data into a model and measuring the response data. The predicted error generated by this extraction becomes the damage-sensitive features that are data that vary according to the change in the damage level. Last, the statistical classification of the parameters is to be made by implementing algorithms that analyze the distributions of the extracted features. According to Figueiredo [8] these algorithms fall into three main groups: classification, regression, and outlier detection. The ideal algorithm for each application depends on the ability to perform supervised or unsupervised learning. Supervised learning can be deployed if examples of damage and undamaged data are available. If only undamaged data examples are known, unsupervised learning is the recommended type of algorithm.

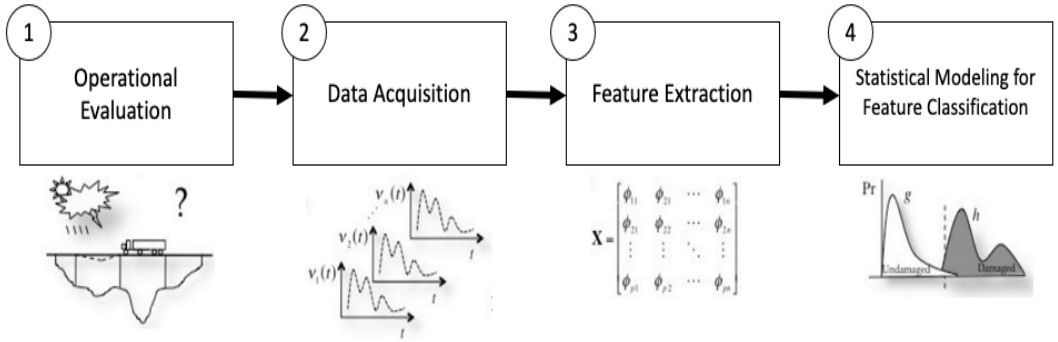


Figure 1.2: SPR paradigm for SHM, adapted from [8]

1.2 Literature Review

Several types of structures may be subjected to changes in their nominal operational status and possible failures. Static structures such as buildings or bridges are constantly under monitoring since their safety and reliability have to be improved, and if there are any damages to their structures, it has to be detected before it reaches a critical state. Over the years, some studies have been developed on data classification using different approaches to obtain features and classify them. This past literature guided the research in terms of methods choice and evaluation criteria.

Figueiredo et al. [9], in 2009, studied four different methods for extracting linear features and obtained an optimal linear model for the three-story building problem. Among the methods were Akaike Information Criterion (AIC), Partial Autocorrelation Function (PAF), Root Mean Squared Error (RMSE), and Singular Value Decomposition (SVD); concluding that autoregressive models (AR) of orders 5, 15, and 30 can represent the behavior of the proposed system adequately.

Figueiredo [8], in another paper publisehd in 2010, applies machine learning tools to classify the obtained linear features. This work uses four kernel-based algorithms to detect damage under varying operational and environmental conditions. Among the methods used are Auto-associative neural network (AANN), Factor Analysis (FA), Mahalanobis Squared Distance (MSD), and SVD, concluding that in terms of general performance, the MSD-based algorithm proved to be the best approach with the lowest type I and II error rates.

The next year Li et al. [10] utilized damage pattern changes in frequency response functions (FRFs) and artificial neural networks to identify damage states. To extract the damage-sensitive features, PCA was the chosen technique.

Nguyen et al. [11] used a method based upon the Monte Carlo simulation methodology to assess the condition of output data obtained from the autoregressive model. In this work, the determination of the order of the autoregressive model was by RMSE.

Santos et al. [12] applied other different tools for the classification of the features, including One-Class Support Vector Machine (One-class SVM), Support Vector Data Description (SVDD), Kernel Principal Component Analysis (KPCA), and Greedy Kernel Principal Component Analysis (GKPCA). The conclusion was that the proposed methods have better classification performance when compared to methods previously used (AANN, FA, MSD, and SVD) due to lower classification errors (Type I and Type II).

Gui et al. [13] used two types of feature extraction methods: the autoregressive model and the residual errors of the statistical parameters. This paper describes some methods to determine the SVM parameters: grid search method, particle swarm optimization, and genetic algorithm. SVM was the chosen method to classify the extracted data, concluding that the three ways had good performances, although the genetic algorithm-based SVM had a better prediction than the others.

Pan [14], in his work, developed a feature extraction method based on Singular Value Decomposition (SVD) by designing a Hankel matrix to enhance

multivariate analysis comparing to other traditional feature extraction methods such as autoregressive model (AR) and multivariate vector autoregressive model (VAR).

More recently, in 2020, Zhang et al. [15] used the deep learning technique, a compact model of a convolutional neural network (CNN), to identify porosity during a 6061 Aluminum alloy welding process.

When it comes to machinery and equipment, the rotating parts are the ones that need good attention and monitoring. Rotating machines are ubiquitously used to transform and transmit power, being essential equipment for monitoring in manufacturing [16]. Devices such as milling [17] and precision machining [18] may be indirectly monitored using vibration measurements, as the modes and frequency spectra are changed in consequence of faulty conditions [19]. However, such signals are often difficult to analyze, requiring automatic data-driven monitoring tools such as machine learning [20, 21, 15, 22]. In some cases, it is necessary to make it possible for complex models to run on embedded hardware platforms to perform online diagnosis at high-frequency rates, such as the NVidia Jetson, a GPU-based ©platform. Devices like this can enable applications that should run decentralized, at high-frequency rates, with relatively complex models that will, in turn, allow more accurate monitoring and decision making in the context of edge computing.

Back in 2011, Moura et al. [23] performed an evaluation of fault recognition efficiency for various combinations of signal processing and pattern recognition techniques combined to diagnose the severity of bearing faults using PCA and ANN.

Abdeljaer et al. [24] used a structural damage detection system using 1D Convolutional Neural Networks (CNNs) that was able to to fuse both feature extraction and classification blocks into a unique learning body.

Mittal [25] in his paper provided a survey on works that evaluate and optimize neural network applications on the Nvidia Jetson platform. Giubilato et al. [26] show that embedded GPU-based performance gains in terms of robustness, resource utilization, and processed frames per second when it comes to the use of a Nvidia Jetson TX2 board instead of a standard workstation-grade computer for Visual Simultaneous Localization and Mapping (SLAM).

Pan et al. [27] proposed a two-stage prediction method based on extreme learning machine (ELM) in order to classify data and predict the remaining useful life of rolling-element bearings. Pinto de Aguiar et al. [28] performed an experimental benchmark between NVIDIA's Jetson Nano and Google's USB Accelerator to use Deep Learning to solve a feature extraction issue in the vineyard context.

More recently, in 2021, Verma et al. [29] used a triaxial MEMS accelerometer (ADXL-345) interfaced with a Raspberry pi 3 (edge device) in order to evaluate an edge-cloud performance for IoT based machinery vibration monitoring.

Based on the foregoing, some methods were chosen to conduct this research: PCA and Autoencoder to perform dimensionality reduction and Logistic regression, support vector machine, decision tree, k-nearest neighborhood, random forest and neural networks to perform the classification of the data.

1.3 Objective

1.3.1 General objective

The objective of this dissertation involves applying SHM statistical procedures for feature extraction and classification to detect damage to structures. For this, the techniques are used on two datasets measured from a three-story building laboratory bench [8] and a slender experimental rig [30].

1.3.2 Specific objectives

For the first case, the following specific objectives are to be achieved:
- Perform linear feature extraction of an AR model;
- Reduce the dimensionality of the features matrix;
- Perform the binary classification of the full data, PCA reduced data and each individual channel using the proposed methods;
- Analyse the results comparing model size and accuracy results.

And for the second case, these are the specific objectives:
- Reduce the data dimensionality using PCA and Autoecoder;
- Perform the classification of the data using the proposed methods;
- Analyse the results comparing accuracy results, inference time (in seconds) and model size.

1.4 Contributions and Publications

This dissertation considers the context of the SPR paradigm contributing to steps three and four of the paradigm.

The first contribution, entitled "Establishing Compromise between Model Accuracy and Hardware Use for Distributed Structural Health Monitoring," published in the XV Simpósio Brasileiro de Automação Inteligente - SBAI Annals, proposes the use of a well-known input/output data-based mathematical model, the AR model to estimate the damage-sensitive features, assuming that the structure presents a linear behavior in its undamaged condition. This work demonstrates that the use of AR features of different sensors distributed along the structure provides the best results in terms of shear accuracy when using the most commonly used shallow models and that it is possible to obtain much smaller models when using dimensionality reduction methods before creating the supervised models without sacrificing the model accuracy significantly. More specifically, using the principal components of the feature space composed of the AR parameters of four accelerometers as features, a decrease in the size of the best model was observed by 27,15%. In contrast, the overall accuracy of the model shrank by only 1,64%. It is important to highlight that the size of the models is essential as, in general, smaller models tend to generalize better, but also, maybe more importantly, smaller models are easier to deploy and maintain in embedded hardware setups. Additionally, concerning the use of a single measurement for SHM, it is shown that (i) using sensors closer to the damage location increases the accuracy, and (ii) using more than one sensor, even if it is far from the damage, can increase the accuracy of damage detection. This highlights the importance of the present work. It shows that it is necessary to devise methods that can orchestrate many sensors simultaneously while keeping the size of the model compact to enable embedded and distributed solutions.

When it comes to rotating machines, there is the second contribution entitled "Rotating Machines Vibration Monitoring with Deep and Shallow Learning on Nvidia Jetson Embedded Computers" that was submited for publication in Manufacturing Letters (MFGLET). This work is concerned with monitoring rotating machines subject to increasing friction while under faulty states, using a GPU-based platform guaranteeing real-time characteristics. We use an experimental setup with rotating inertias and a low-stiffness slender connecting rod to represent rotating machinery responses to different abnormal conditions. The embedded hardware is employed using unsupervised learning, leading to new research directions in rotating equipment monitoring using complex data-driven modeling paradigms.

1.5 Document structure

This dissertation is organized into the following chapters:

- Chapter 1 - Introduction: introduces the dissertation, its motivation, and goals; briefly reviews the literature related to machine learning techniques applied to structural health monitoring cases and points out the contribution of this work.

- Chapter 2 - Theoretical Background: presents a theoretical review regarding systems models, dimensionality reduction techniques, feature extraction techniques, statistical algorithms for feature classification, and machine learning methods;

- Chapter 3 - Establishing compromise between model accuracy and hardware use for distributed structural health monitoring: focus on describing the contributions, process, and results obtained from the Three-Story Building case study;

- Chapter 4 - Rotating Machines Vibration Monitoring with Deep and Shallow Learning on Nvidia Jetson Embedded Computers: focus on describing the contributions, process, and results obtained from the Rotational Test Rig case study;

- Chapter 5 - Conclusions: summarizes the conclusions based on the results obtained and lists recommendations for future work.

2
Theoretical Background

In this chapter, feature extraction, classification, and validation methods used in this work are devised.

2.1
Feature extraction and dimensionality reduction models

The Autoregressive model was chosen for feature extraction since AR models can be used as damage-sensitive feature extractors based on the AR parameters (used in this work) or residual errors. [9]

2.1.1
Autoregressive Model (AR)

The autoregressive model AR(p) is a statistical model that represents a process. It was first applied by Klein on 1997 on Yule's 1927 analysis of the time-series behavior of sunspots [31]. This model was considered to obtain the linear parameters with a total of $p = n_a$ parameters to estimate, being p the model order, disregarding the input data of the system. It can be written in (2-1), being x_i the measured signal at time t_i. The ε_i term refers to the residual error at the sampling instant i. It can be written as given in (2-2).

$$x_i = \sum_{j=1}^{p} \phi_j x_{i-j} + \varepsilon_i \tag{2-1}$$

$$\varepsilon_i = x_1 - \hat{x}_i \tag{2-2}$$

being $\hat{x}_i$ the predicted measure at sampling instant i. The parameter ϕ_j is estimated using batch least-squares approaches or Yule-Walker equations [32].

2.1.2
Principal Component Analysis (PCA)

Some methods are required to reduce its dimensionality in an interpretable way preserving most information to interpret large datasets [33]. One of the oldest methods to do such a thing is the principal component analysis (PCA). It was first proposed by Karl Pearson in 1901 and later renamed by Harold Hotelling in 1933 [34]. Mainly, it performs the pre-processing of the

data by mean subtraction and setting variance to 1 before performing singular value decomposition [35]. It computes the mean matrix as given in (2-3) where $\bar{x}$ is the row-wise mean, calculated in (2-4).

$$\bar{X} = \begin{bmatrix} 1 \\ \dots \\ 1 \end{bmatrix} \bar{x} \tag{2-3}$$

$$\bar{x}_j = \frac{1}{n} \sum_{i=1}^{n} X_{ij} \tag{2-4}$$

Subtracting the mean matrix from the large matrix X results in the mean-subtracted data B given in (2-5).

$$B = X - \bar{X} \tag{2-5}$$

The first principal component is given

$$u_1 = argmax\ u_1^* B^* B u_1;\ \|u_1\| = 1, \tag{2-6}$$

being the eigenvector of B^*B the largest eigenvalue [35].

It is also possible to obtain the principal components by computing the eigenvalue decomposition of the covariance matrix (C), as given in (2-7) and C is calculated in (2-8) [35].

$$CV = VD \tag{2-7}$$

$$C = \frac{1}{n-1} B^* B. \tag{2-8}$$

Being C the covariance matrix, V the matrix eigenvectors of C and D the diagonal matrix of all eigenvalues of C.

2.2 Machine learning models

This section presents the machine learning methods chosen for this application. All machine learning methods chosen to perform the classification of the extracted features are supervised methods and are the following.

2.2.1 Support Vector Machine (SVM)

Support Vector Machines are potent methods for performing classification of small to medium-sized datasets. It was proposed by Boser et al. [36]. It is a supervised learning algorithm that aims to classify a set of data points mapped to a multidimensional characteristic space using a kernel function.

A training vector in two classes given as $\mathrm{x_i} \in \mathrm{R^n}$, $\mathrm{i} = 1, ..., \mathrm{l}$ and an indicator vector $\mathrm{y} \in \mathrm{R^l}$ as $\mathrm{y_i} \in \{1, 1\}$ [37], SVM solves the primal optimization problem as following

$$\begin{aligned} \min w, b, \xi \frac{1}{2} w^T w + C \sum_{i=1}^{l} \xi_i \\ \text{subject to } y_i \left(w^T \phi\left(x_i\right) + b \right) \geq 1 - \xi_i, \\ \xi_i \geq 0, \ i = 1, ..., l \end{aligned} \tag{2-9}$$

Where $\mathrm{C} > 0$ is the regularization parameter and $\phi\left(\mathrm{x_i}\right)$ maps $\mathrm{x_i}$ int a higher dimensional space. The main goal is to find $\mathrm{w} \in \mathbb{R}^\mathrm{n}$ and $\mathrm{b} \in \mathbb{R}$ so the prediction given by $\mathrm{sign}(\mathrm{w^T}\phi\left(\mathrm{x_i}\right) + \mathrm{b})$ is correct for the majority of the samples. The result for the part $\mathrm{y_i}\left(\mathrm{w^T}\phi\left(\mathrm{x_i}\right) + \mathrm{b}\right))$ is ideally ≥ 1 for all samples indicating perfect prediction, but not all cases are perfectly separable, so the algorithm allow some samples to be distant in ξ_i from their correct margin boundary.

The vector variable w can have higher dimensionality, and this problem is solved in (2-10). After the problem solving, the output decision function is given in (2-11), and its sign corresponds to the predicted class.

$$\begin{aligned} \min \propto \frac{1}{2} \propto^T Q \propto - e^T \propto \\ \text{subject to } y^T \propto = 0, \\ 0 \leq \propto_i, \ i = 1, ..., l \end{aligned} \tag{2-10}$$

$$w = \sum_{i=1}^{l} y_i \propto_i K\left(x_i, x\right) \tag{2-11}$$

Where e is a vector of ones, Q is a $\mathrm{l} \times \mathrm{l}$ positive semi definite matrix, $\propto_\mathrm{i}$ are the dual coefficients upper-bounded by C and $\mathrm{K}\left(\mathrm{x_i}, \mathrm{x}\right)$ is the kernel given by (2-12).

$$K\left(x_i, x_j\right) = \ \phi\left(x_i\right)^T \phi\left(x_j\right) \tag{2-12}$$

2.2.2 Logistic Regression (LGR)

Logistic Regression is one of the most common methods used for binary data response. It was first proposed in the 19th century to describe the growth of population and the course of chemical reactions [38]. The model takes the natural logarithm of the odds as a regression function of the predictors being the odds the ratio of the probability of the event happening and the probability of the event not happening [39]. It will model the probability based on individual characteristics, which is given by

$$\log\left(\frac{\pi}{1-\pi}\right) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_m x_m \tag{2-13}$$

where π is the probability of the event, β_i is the regression coefficient, and $\mathrm{x_i}$ the explanatory variable [40].

2.2.3 Decision Tree Classifier (DTC)

It was proposed by Breiman et al. [41] in 1984 motivated by a Bayesian model. A decision tree is a structure to express a sequential classification process [42]. According to Rokach and Maimon [43], a decision tree is formed by nodes that can be "root" nodes that have no incoming edges and "internal or test" nodes that have outgoing edges. All other nodes are called leaves of the tree. Each internal node splits into two or more sub-spaces, and each leaf is assigned to one class representing the ideal target value. Some criteria can be used to measure the quality of the tree. Two main methods were used: the Gini index and information gain. The Gini Index measures divergences between the probability distributions of the target attribute's values and is defined in (2-14) [43].

$$Gini\,(y, S) = 1 - \sum_{c_j \epsilon dom(y)} \left(\frac{\left|\sigma_{y=c_j} S\right|}{|S|}\right)^2 \tag{2-14}$$

Where S is the training set and y is the probability vector of the target attribute. The evaluation criterion for selecting the attribute $\mathrm{a_i}$ is given in Eq. (2-15).

$$GG\,(a_i, S) = Gini\,(y, S) - \sum_{v_{i,j} \epsilon dom(a_i)} \frac{\left|\sigma_{a_i=v_{i,j}} S\right|}{|S|} \cdot Gini\left(y, \sigma_{a_i=v_{i,j}} S\right) \tag{2-15}$$

Where GG is the Gini Gain. On the other hand, another univariate criterion to decide the best attribute to split is the impurity-based criterion that uses the entropy method as an impurity measure.

$$IG\left(a_i, S\right) = E\left(y, S\right) - \sum_{v_{i,j} \epsilon dom(a_i)} \frac{\left|\sigma_{a_i = v_{i,j}} S\right|}{|S|} \cdot E\left(y, \sigma_{a_i = v_{i,j}} S\right) \tag{2-16}$$

where E stands for entropy and is calculated in (2-17).

$$E\left(y, S\right) = \sum_{c_j \epsilon dom(y)} -\frac{\left|\sigma_{y = c_j} S\right|}{|S|} \cdot \log_2 \frac{\left|\sigma_{y = c_j} S\right|}{|S|} \tag{2-17}$$

The search for a split won't stop until at least one valid partition of the node samples is found.

2.2.4 K-nearest Neighbourhood (KNN)

Proposed by Evelyn Fix and Joseph Hodges [44] in 1951 and later expanded by Thomas Cover [45] in 1967. The KNN method searches for groups of K objects in the closest training data to similar objects in test data and based on the distance the K nearest neighbors identified and classified [42]. The Euclidian distance is one typical distance metric and is given by (2-18).

$$d\left(p, q\right) = \sqrt{\sum \left(pi - qi\right)^2} \tag{2-18}$$

2.2.5 Random Forest Classifier (RFC)

A random forest classifier, proposed by Breiman [46] in 2001, is an assembly of several decision trees, generally trained via the bagging method. It creates random decision trees, gets the prediction of each tree, and selects the best solution through voting [47]. A scheme of a random forest classifier is illustrated in Fig. 2.1.

In this classifier, a random vector Θ_k is produced with the same distribution but independent of the past random vectors independent of the past random vectors Θ_1, ..., Θ_{k-1}. Each tree is grown using a training set and a random vector Θ_k resulting in a classifier $\{h(x, \Theta_x), k = 1, \cdots\}$ at input vector x [46].

The generalization error is given by Equation (2-19) where X and Y are random vectors that indicate that the probability is over X,Y space. The marin function mg measures the extent to which the average number of votes at the random vectors exceeds any other class's average vote.

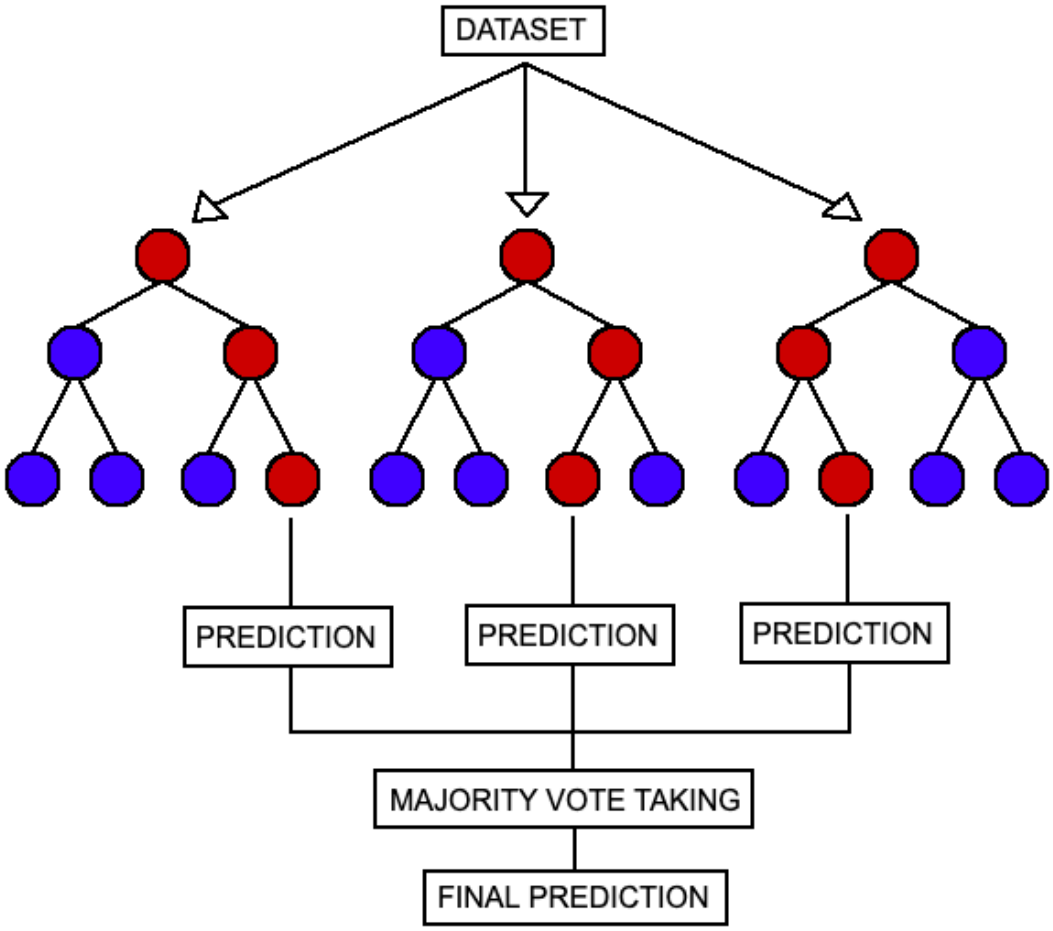


Figure 2.1: Workflow of a Random Forest Classifier, adapted from [48]

$$PE^* = P_{X,Y}(mg(X,Y) < 0)) \tag{2-19}$$

The margin function is defined in Equation (2-21) where I is the indicator function [46].

$$mg(X,Y) = av_k I(h_k(X) = Y) - max_{j \neq Y} av_k I(h_k(X) = j) \tag{2-20}$$

In this particular case, the random forest consists of 100 trees, and the forest chooses a class considering the most out of 100 votes.

2.2.6 Feedforward Neural Network (FFANN)

Neural networks started as a very popular machine learning algorithm named perceptron and it was proposed by Rosenblatt [49] in 1958. An artificial neural network (ANN) is a group of interconnected mathematical modules called artificial neurons or nodes. These artificial neurons apply a mathematical activation function to the sum of the weighted inputs. The outputs generated by this action may be connected to a bias and produce a sequence of real-valued activation's [50]. They are usually organized in layers, and it consists of the input, output, and the hidden layers [19]. The hidden layers are the ones between the input and output.

ANNs are said to be shallow when they have only a single hidden layer and many neurons. On the other hand, deep networks are said to be composed of more than one hidden layer. If there is no feedback from the outputs towards the network's inputs, it is referred to as a feed-forward neural network. The FFANN structure is illustrated in Fig. 2.2.

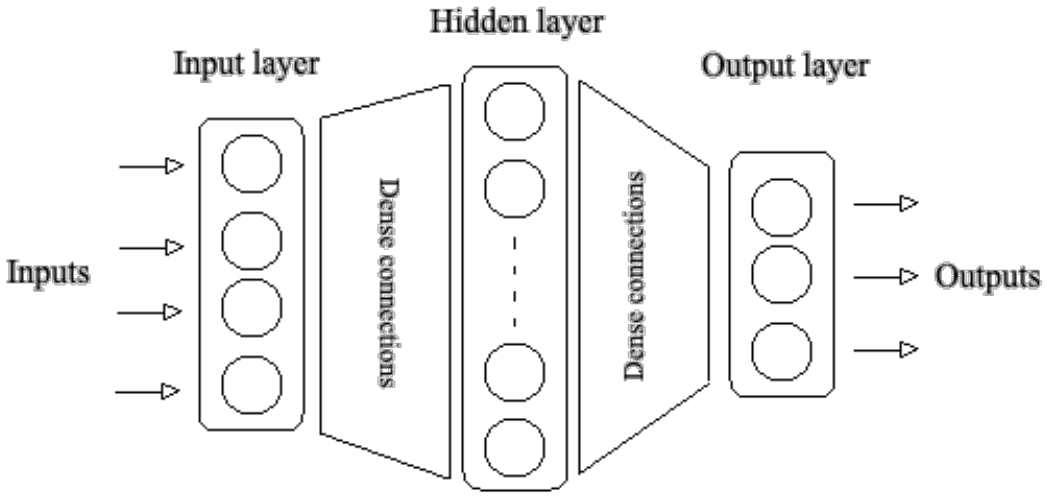


Figure 2.2: Feed-forward Artificial Neural Network structure with one hidden-layer, adapted from [51]

2.2.7 Deep Autoencoder (DAE)

An autoencoder is a type of neural network. It consists of an encoder and a decoder connected in serial, each owning a single hidden layer. It learns the intrinsic network features by reconstructing the original network at its output layer [52].

The main difference between a traditional autoencoder and a deep autoencoder is that the DAE contains more than just one hidden layer in its net [53], as illustrated in Fig 2.3. It learns a more robust feature representation in an unsupervised way [52]. It allows an effective feature extraction through hierarchical nonlinear mapping when multiple hidden layers are involved, resulting in a reduction of training dataset [53].

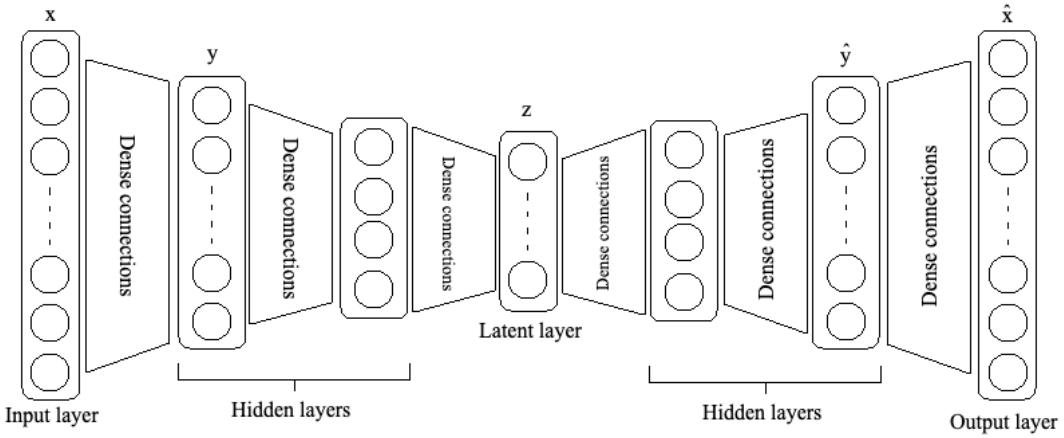


Figure 2.3: Deep autoencoder architecture, adapted from [54]

The encoder module The loss function can be expressed as in (2-21) for an unlabelled dataset X = [x_1, x_2, x_3,$\cdots$,x_n]:

$$L = f(\phi : X, X) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{1}{2}\left\|\hat{x}_i - x_i\right\|^2\right) + \lambda(\phi) \tag{2-21}$$

2.3 Hyperparameters Optimization

There are two variables that are important when it comes to machine learning algorithms [55]: Parameters that are values that the algorithm tunes according to the provided data set and the hyperparameters that are parameters defined before the training step that defines how the training of the model is supposed to happen.

The choice of these parameters affects directly the performance of the model, and there is not much guidance on how to choose the hyperparameters, that's why it is important to tune or optimize this values. Optimize these hyperparameters means to find a combination of hyperparameters that performs best when measured in a validation set.

This optimization can be done by some approaches such as grid search, random search and Bayesian search. This research uses random search for the

optimization of the hyperparameters. Fig 2.4 illustrates two approaches, grid and random search, showing that random search explores the hyperparameter space more widely. The optimal point, represented by the blue circle found by random search stand for a more accurate model than the one in grid search even though they both have the same number of trials [56].

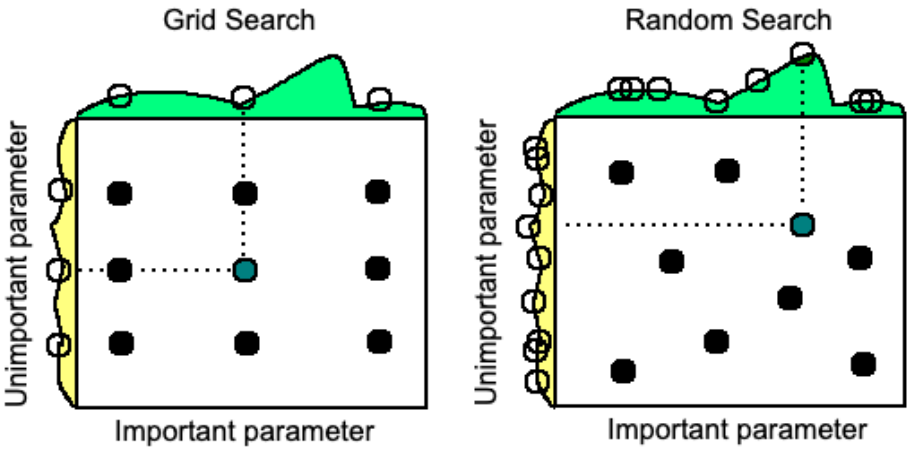


Figure 2.4: Comparison between grid search and random search for hyperparameter optimization, adapted from [56]

Random search randomly picks hyperparameters and trains the model using a training set. This approach consumes less time and resources [55] since it doesn't depend on the results of previous training jobs. This way, a maximum number of concurrent training steps can be performed without affecting the performance of the search.

In random search, the number of trials is defined in order to set the number of sets of hyperparameters to be tried [55].

Another type of hyperparameter optimization method is Adam optimization algorithm. It was proposed by Kingma and Ba [57] in 2014 and, according to them, its is computationally efficient and has little memory requirement. It is well suited for problems that are large in terms of data. This optimization method is a stochastic gradient descent method, based on adaptive estimation of first and second-order moments [57].

2.4 Model Validation

The validation process is essential to guarantee the generalization of a machine learning model to decide which model performs best.

2.4.1 Cross Validation

Cross-validation is a resampling approach, a statistical technique used to test a machine learning model's ability to generalize an independent data set or data not used to train the test set.

Just a limited amount of data is available in most cases [58]. That is why splitting the data is needed, so part of the data is used for training the algorithm (training set), and the other part is used to evaluate the algorithm's performance (validation set). The model's training is often done within an iterative process. In each iteration, the performance is measured to update the model parameters for error reduction when applied to the data in the training set. Therefore, test data should not be used during model training and tuning. Otherwise, the generalization error would be unreliable [59].

Cross-validation can be used for several applications, such as regression and classification. For this research, aiming for better data classification, the Monte Carlo Hold-out Cross Validation was chosen.

2.4.2 Monte Carlo Hold-out Cross-Validation

It was proposed by Picard and Cook [60], and, according to Lendasse et al. [61] in this validation method, the data is randomly divided into several train and validation sets. According to Xu and Liang [62], this process is repeated N times (N = 1,2,3, ..., N) and is defined by (2-22)

$$MCCV_{n_v}(k) = \frac{1}{Nn_v} \sum_{i=1}^{N} \|y_{S_v(i)} - \hat{y}_{S_v(i)}\|^2 \tag{2-22}$$

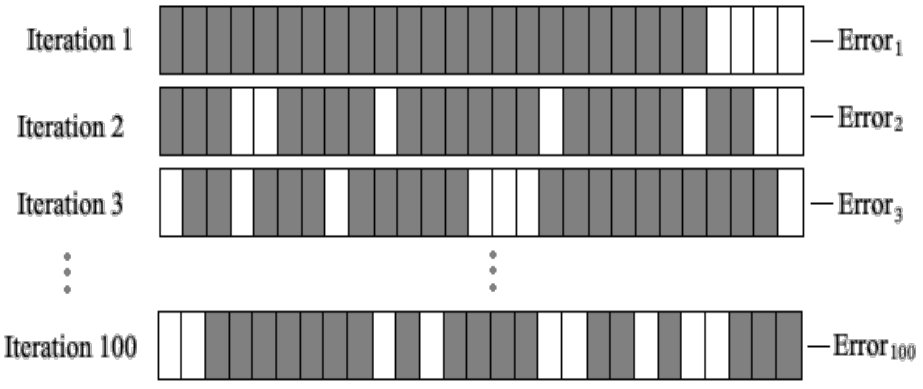


Figure 2.5: Monte Carlo Cross-Validation with 100 iterations architecture

where $n_v = n - n_c$ samples for the validation model, n_c is the samples for the fitting model, S_v corresponds to the samples of the validation sets.

Figure 2.5 illustrates the Monte Carlo Cross Validation dynamics. Data is randomly splitted, being the train set represented in gray and the test set represented in white. This s, the train-test split percentage vary from iteration to iteration. The model is then fitted on the train data set for each iteration and a test error is calculated. After the N iterations, the average of the errors is taken using Equation (2-23). This process is usually known as the evaluation of residuals.

$$Error = \frac{1}{N}\sum_{i=1}^{N} Error_i \tag{2-23}$$

3
Establishing compromise between model accuracy and hardware use for distributed structural health monitoring

This chapter introduces the first experimental test-bed structure and its data used to conduct the research and the obtained results.

3.1
Test bed and structure

The first experimental setup approached in this research is a building structure, displayed in Figure 3.1. It consists of a four-degree of freedom three-story building formed by aluminum plates (30.5 x 30.5 x 2.5 cm) and columns (17.7 x 2.5 x 0.6 cm) mounted with bolted joints [9]. This structure was mounted on the rails to allow movement in only one direction (x-direction, as shown in Figure 1). In addition, a column (15.0 x 2.5 x 2.5 cm) is arranged in the center of the plate corresponding to the upper floor to simulate damage, inducing a non-linear behavior when it collides with a bumper mounted on the floor below. The position of the bumper is adjustable to vary the extent of the impact that occurs at a specific excitation level.

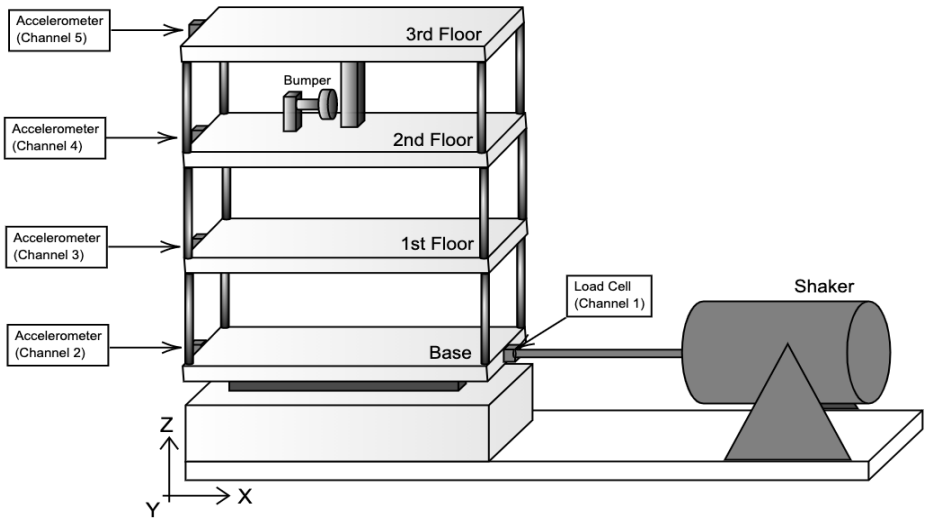


Figure 3.1: Three-story building test bed structure illustration.

3.2 Data Acquisition System

The Los Alamos National Laboratory obtained the data used in this research by load cells and accelerometers. A load cell with 2.2mV/N sensitivity was mounted at the end of a stinger to measure the input force of the shaker that stimulates the structure. Four accelerometers with 1000mV/N sensitivity were mounted in the centerline of each floor on the opposite side of the excitation source. The data acquisition system used to collect the data was a Dactron Spectrabook. This system provides the excitation signal to the shaker since it is connected to a Techron 5530 Power Supply Amplifier.

According to the Technical Report [9], the sensor signals were discretized with 8192 data points sampled at 3.125ms intervals with a sampling frequency of 320Hz, which yielded time histories of 25.6 seconds in duration. The excitation level was set to correspond to 20N RMS measured in Channel 1.

3.3 Dataset

The data considered 17 structural states, described in Table 3.1. Force and acceleration time histories were collected for various structural state conditions.

The structural state conditions are divided into four main groups: baseline condition, which is the reference structural state and is labeled as State 1; the group of conditions where the mass and stiffness of the columns change, labeled as State 2 to 9; the group of conditions where damage state are simulated by introducing nonlinearities using the bumper and the suspended column by varying the gap between them labeled as State 10 to 14 and; the last group that simulates the damages with the bumper and the column plus a mass addition in different floors at a time labeled as State 15 to 17.

The structure was excited ten times for each structural state to consider the variability of the data. Thus, ten-time histories were measured for each of the 17 structural states for all five transducers (850 tests). The data set for the 850 tests performed on the structure consists of an array of rows, columns, and depth. The lines refer to all samples obtained during 25 seconds in each of the 850 tests; the five columns refer to each of the five channels, and the 850 depth columns refer to the 850 tests performed (10 tests for each of the 17 states for each of the five channels).

Figure 3.2 (a) show the acceleration-time history for States 1, 3, 6, 10 and 16. A closer view in time can be seen in Figure 3.2 (b). Analyzing the time history, some amplitude differences can be seen as the situation changes.

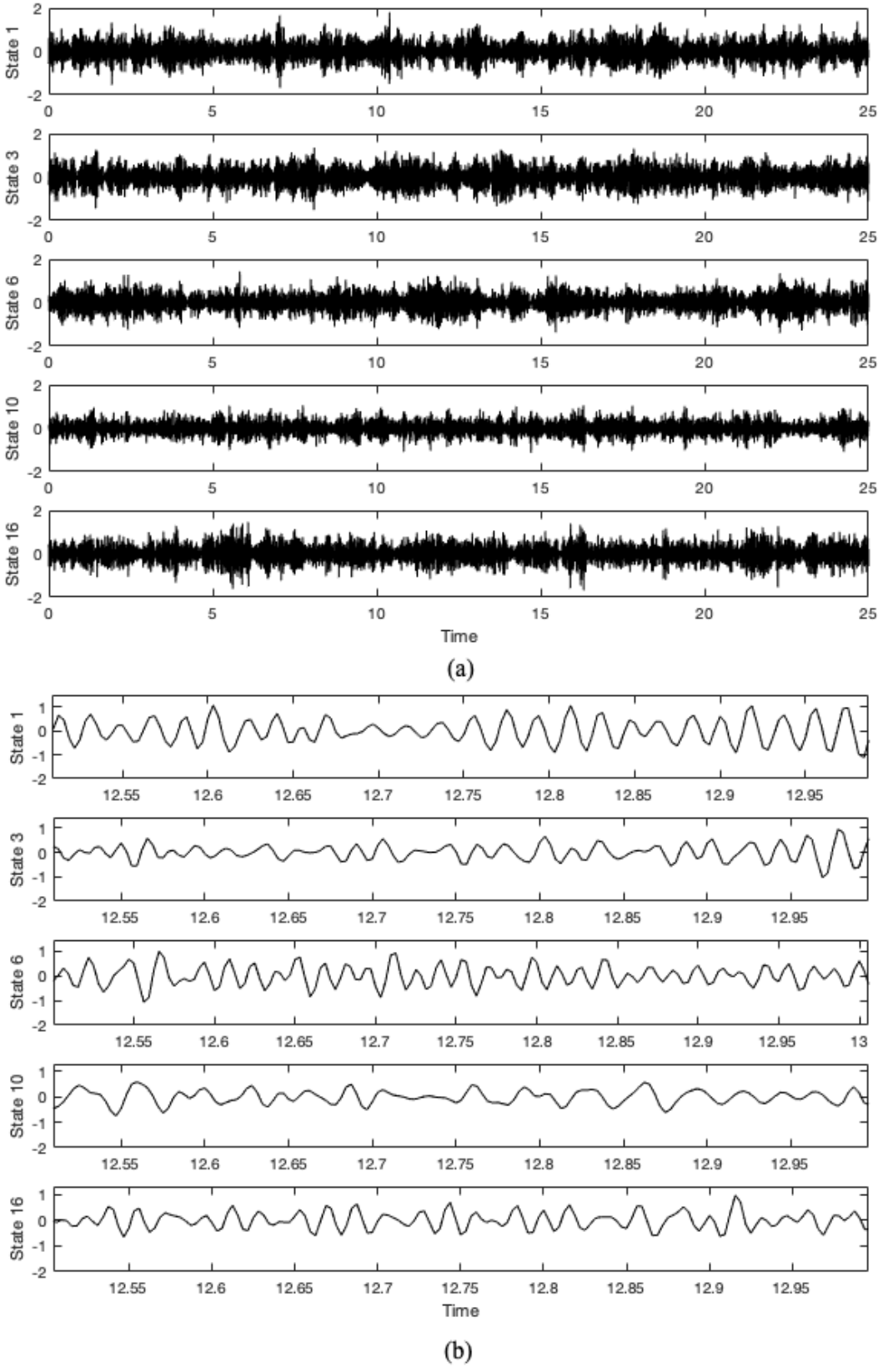


Figure 3.2: Acceleration-time history from Channel 5 in different states (a) and zoom view in time (b).

But it is easier to see the differences and identify any damage to the structure when looking at the FRF plots or the extracted features of the model, which will be explored in the upcoming sections.

Table 3.1: Data labels of the structural state conditions [9]

State	Condition	Description
1	Undamaged	Baseline condition
2	Undamaged	Mass = 1,2kg at the base
3	Undamaged	Mass = 1,2kg at the 1st floor
4	Undamaged	87.5% stiffness reduction in column 1BD
5	Undamaged	87.5% stiffness reduction in column 1AD and 1BD
6	Undamaged	87.5% stiffness reduction in column 2BD
7	Undamaged	87.5% stiffness reduction in column 2AD and 2BD
8	Undamaged	87.5% stiffness reduction in column 3BD
9	Undamaged	87.5% stiffness reduction in column 3AD and 3BD
10	Damaged	Gap = 0.20 mm
11	Damaged	Gap = 0.15 mm
12	Damaged	Gap = 0.13 mm
13	Damaged	Gap = 0.10 mm
14	Damaged	Gap = 0.05 mm
15	Damaged	Gap = 0.20 mm and 1.2 kg mass at the base
16	Damaged	Gap = 0.20 mm and 1.2 kg mass at the 1st floor
17	Damaged	Gap = 0.10 mm and 1.2 kg mass at the 1st floor

3.4 Results and Analysis

This section will show the results obtained from the dimensionality reduction of the model and the binary classification results of the data. According to Figueiredo et al. [32], for this particular case, the optimal order stands between 15 to 30. These orders allow discrimination between the undamaged and damaged states when all conditions proposed in Table 3.1 are considered. For this paper, a model of order 30 is constructed, generating, for each channel, one 850 x 31 matrix of parameters. As the features are extracted and plotted in a graph, it is clear to identify and separate the undamaged data from the damaged data. Fig. 3.3 show some of the Channel 5 features plots for undamaged states 1, 3, and 6 and damaged states 10 and 16. It suggests that the more nonlinearities introduced to the structure, the more decreased the features' amplitude. In the upcoming subsections, results of the dimensionality reduction and separation of the between damaged and undamaged states are shown, confirming it can perform with good accuracy compared to the full data.

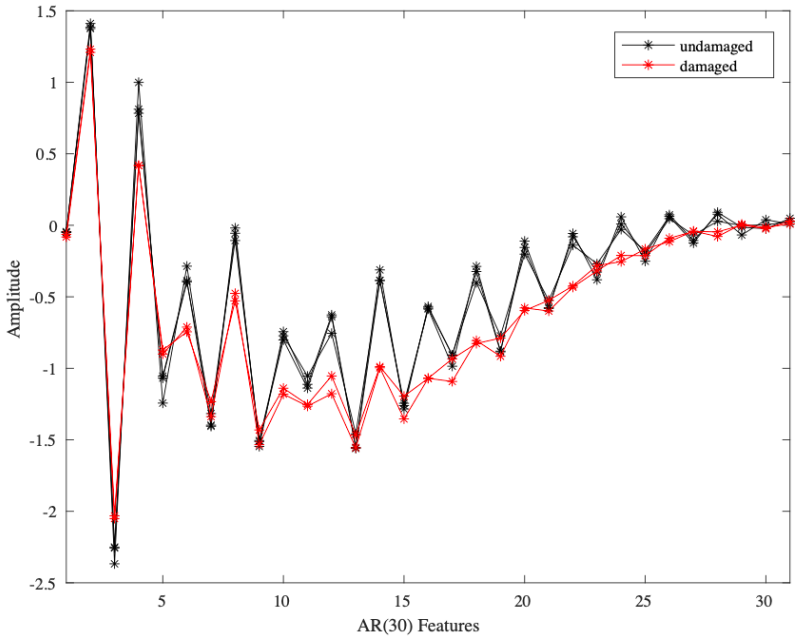


Figure 3.3: Channel 5 AR(30) features.

3.4.1 Matrix concatenation and model accuracy analysis

The PCA algorithm was designed to retain 99% of the data variability to perform the dimensionality reduction of all concatenated data. That resulted in a model containing 11 principal components, as illustrated in Figure 3.4. It resulted in a matrix of 850 x 11 parameters with 99,23% of explained variance. This smaller matrix in dimensions and byte size can perfectly describe the system and be well classified using the proposed machine learning techniques.

For the classification step using Monte Carlo Hold-Out Cross-Validation, a hundred models were created randomly for training and test stages from all parameter matrices obtained during the feature extraction step, using a 50/50 ratio for both the test and training sets. Some hyperparameters were chosen manually for each model to guide the learning process and are listed in Table 3.2. The hyperparameters were later optimized to get the most out of the classification process. The optimization algorithm used in this case was the random search that performs a randomized search on hyperparameters to choose the one that performs best.

LGR classifier implements logistic regression using liblinear as the algorithm used in the optimization problem. It supports two binary linear classifiers: LGR and SVM [63]. The C value parameter describes the inverse of regularization strength. The smaller the value, the stronger the regularization. The kernel is the main hyperparameter of an SVM model. Linear, polynomial,

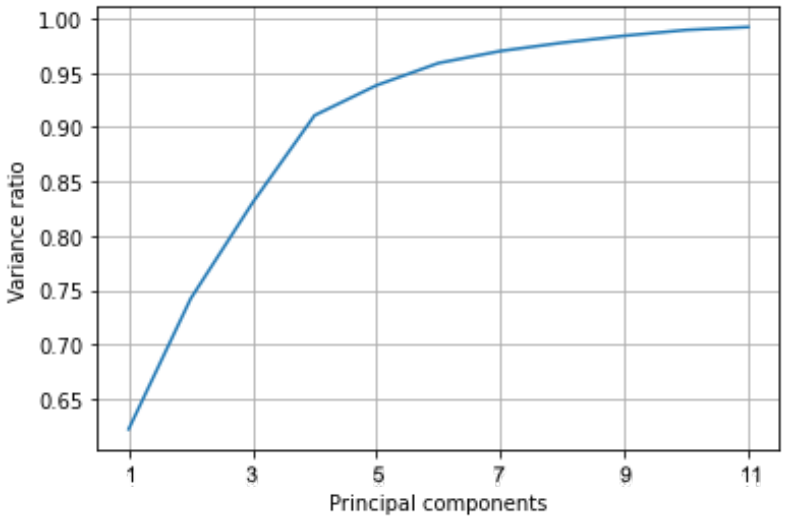


Figure 3.4: Explained variance ratio vs. total principal components.

radial, and sigmoid kernel functions were set to avoid complex calculations. Some other parameters must be specified when kernel types are defined, such as C values for linear kernel functions, the degree for polynomial functions, and the kernel coefficient (γ) for radial, polynomial and sigmoid kernel functions.

Table 3.2: Model hyperparameters

LGR	
C values	Between 10^{-1} and 10^{3}
Solver	"liblinear"
SVM	
C values	Between 10^{-1} and 10^{3}
Kernel type	“linear”, “poly”, “rbf” and “sigmoid”
Kernel degree	2 to 5
Kernel coefficient	10^{-4} and 10
DTC	
Split criterion	"gini"and "entropy"
Split strategy	"best"and "random"
KNN	
Number of neighbors	Between 2 and 100
RFC	
Split criterion	"gini"and "entropy"
Split strategy	"best" and "random"
Number of trees	Between 2 and 100

When it comes to DTC, the split criterion that measures the quality of the split must be set. Ginni and Entropy were chosen for starts and the strategy used to select the best split were "best" and "random," which selects the best split and the best random split, respectively. For the KNN model, the number of neighbors (K) is the main deciding factor and it was set between 2 and 100, randomly between the iterations.

The accuracy results can be seen in Table 3.3. A better view of the accuracy results is illustrated in Fig. 3.5.

Table 3.3: Accuracy results for every data classification

	All Features	**PCA**	**Ch5**	**Ch4**	**Ch3**	**Ch2**
LGR	0.9965	0.9731	0.9563	0.9549	0.8873	0.8644
SVM	0.9972	0.9863	0.9664	0.9611	0.8897	0.8882
DTC	0.9625	0.9366	0.8915	0.8913	0.7605	0.6540
KNN	0.9874	0.9750	0.9002	0.8982	0.8288	0.6986
RFC	0.9885	0.9722	0.9354	0.9303	0.8388	0.7317

The individual channel classifications perform better from Channel 4 and 5 since their accelerometers are closer to the damage source (bar on the 3rd floor + bumper on the 2nd floor). This pattern follows for all predicted models, where the accuracy results related to the accelerometers farther from the damage source are smaller. The accuracy score from the PCA of all data is only 1,64% (medium) less than the whole matrix, which can be considered acceptable since the accuracy remains at good values, above 90%. Nonetheless, the number of inputs is considerably smaller when using PCA compared to all channels. This will be investigated in the next section.

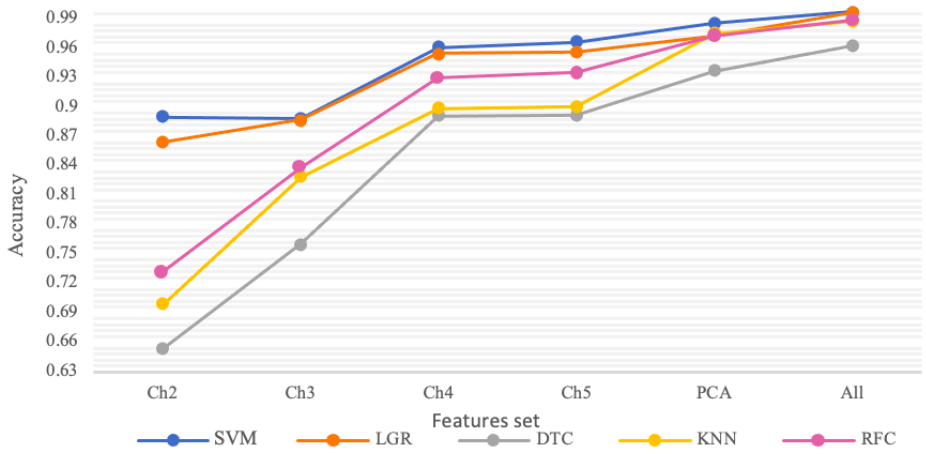


Figure 3.5: Accuracy plots for all machine learning models used for data classification.

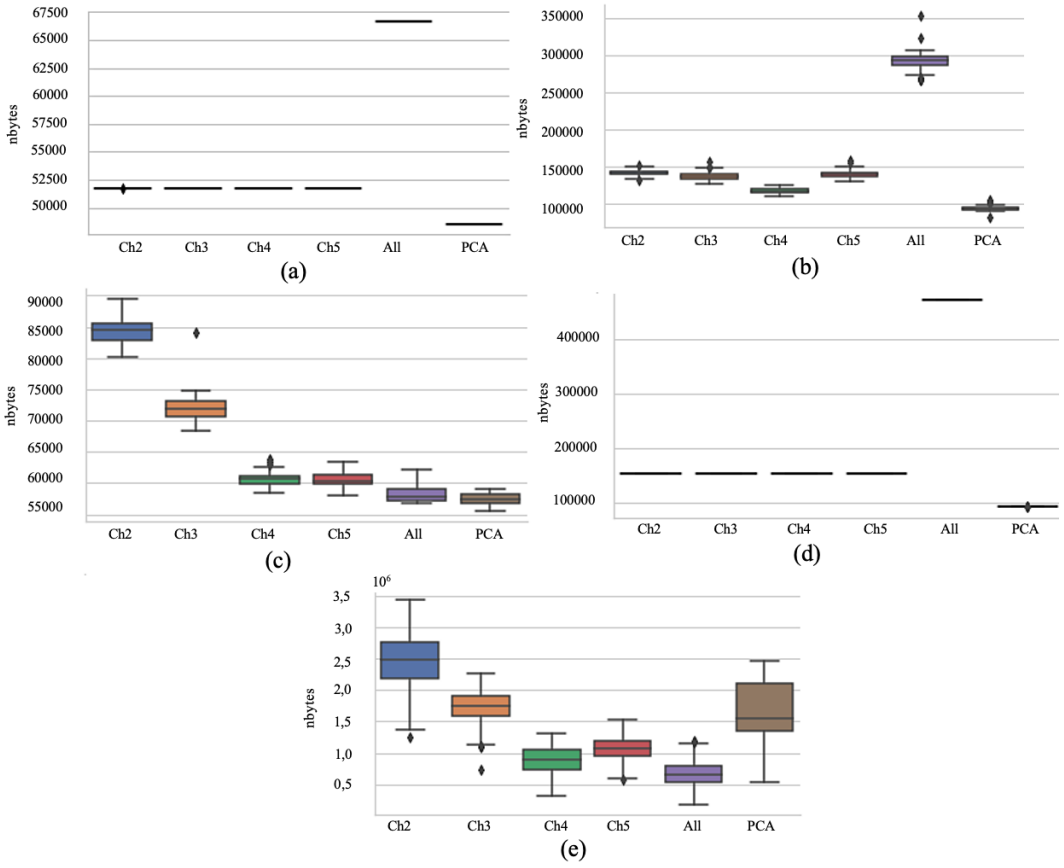


Figure 3.6: Byte sizes of SVM model (a), LGR model (b), DTC model (c), KNN model (d) and RFC model (e)

3.4.2 Model Byte Size Analysis

When it comes to the size in number of bytes of the model, the results of the average number of bytes were calculated using the function "getsizeof" from the system-specific parameters (sys) module in Python and can be seen in Figure 3.6. The mean values of the sizes were concatenated in Table 3.4. Putting these data into a graph to better visualize the data, as illustrated in Fig. 3.7, it is visible that the dimensionality reduction models of the majority of the methods have a lower number of bytes size comparing to the individual channels themselves.

Except for the RFC models, which are the ones with most significant byte sizes, including the model that had its dimensionality reduced, that has above 2Mb in size. This behavior can be explained since the RFC model is more robust when compared to the other models, requiring more space since, in this case, 100 trees are associated to carry out the model's classification.

After performing the dimensionality reduction, the overall byte size of

Table 3.4: Mean bytes numbers result for every data classification model in Mb

	All Features	PCA	Ch5	Ch4	Ch3	Ch2
LGR	0.048	0.066	0.051	0.051	0.051	0.051
SVM	0.281	0.091	0.136	0.120	0.142	0.132
DTC	0.058	0.060	0.060	0.064	0.073	0.087
KNN	0.472	0.092	0.154	0.154	0.154	0.154
RFC	2.464	0.561	1.044	0.645	1.656	1.875

the models is smaller by 27.15% (mean) than all channels concatenated and even the individual channels alone. And, as seen in section 5.1, the accuracy is only 1.64% smaller when comparing the results from all data combined and the PCA data.

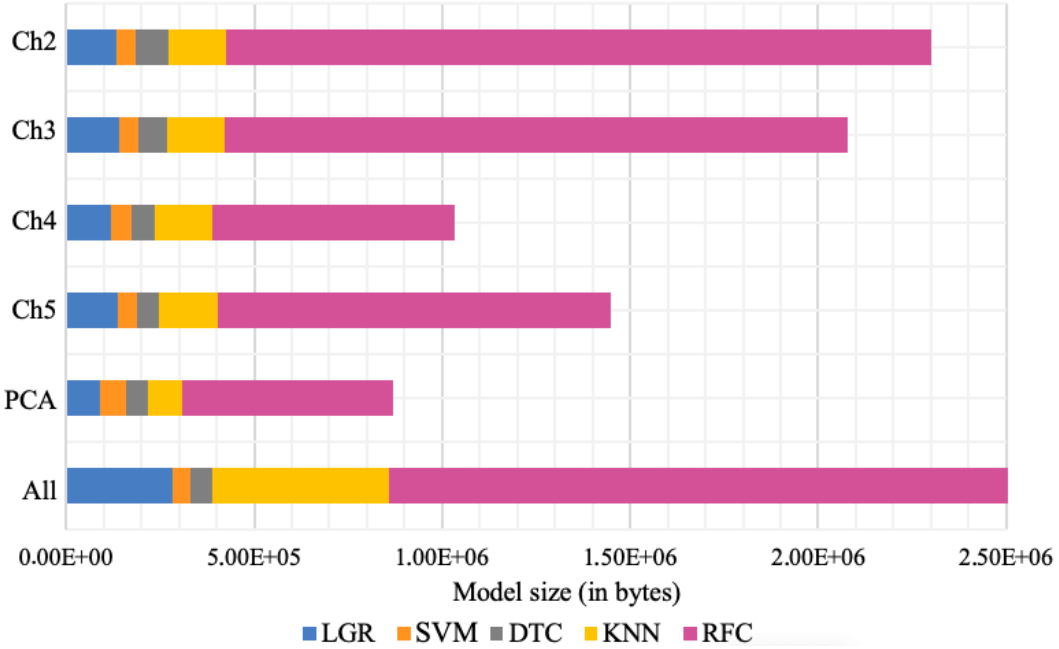


Figure 3.7: Comparison between model sizes

This can be beneficial in terms of having a smaller data set that describes the system's behavior adequately, which reduces computational efforts and reduces the time of execution of the algorithm. Besides, it may cover most cases since larger data sets, like the one in this study case, can eventually have their linear features set reduced to facilitate their deployment.

3.5 Chapter Remarks

Based on linear feature extraction, an approach of dimensionality reduction and several data classifications were performed with Monte Carlo hold out cross-validation. The main idea is to relate the byte size of the full models and the byte size of the dimensionality reduction of the models, verifying the changes in size and accuracy of the results when applying PCA to the extensive linear features data set compared to its original set. A hundred random validation experiments were conducted, and the results prove that the dimensionality reduction of this model was well succeeded in terms of size reduction, good description of the model, and accuracy results of the classification step. Making it reasonable and accurate to work with smaller version models of a larger data set.

4
Rotating Machines Vibration Monitoring with Deep and Shallow Learning on Nvidia Jetson Embedded Computers

This chapter introduces the second experimental test-bed structure, and its data used to conduct the research and the obtained results.

4.1
Experimental Setup Description

The test rig is composed of a DC motor connected to two solid discs by a low-stiffness shaft. The motor is an ENGEL GNM5480-G6.1 DC motor with a planetary gearbox [64]. The shaft transmits rotation to the discs, which are free to rotate, and bearings constrain lateral motions. Besides, we may independently apply resistive torques to the discs. There are two braking devices, consisting of pins that pass through the bearing support and touch the discs, as shown in Fig 4.1. The dry contact between the pins and the discs produces friction torque, leading the system to exhibit torsional vibrations. The system may exhibit torsional vibration without friction when the natural frequencies are excited. The dry friction may originate stick-slip phenomena, in general undesirable as it implies energy storage in elastic elements, which may deteriorate once it is released as kinetic energy. Thus it is essential to monitor friction conditions in rotating machinery.

We measure the angular displacement measurements of the discs and the motor using three LS Mecapion H40-8-1000VL encoders. The encoders are optical quadrature type and have a resolution of 1000 ticks per revolution, and we use these measurements to obtain the angular velocities by differentiation. Load cells S10 R9 255 from Kratos equipment and SV50 R-5 from Alpha Instruments measure the normal contact forces between pins and discs. Additionally, we utilize the National Instruments cDAQ- 9174 USB with four slots to perform the data acquisition and recording, with a mean sampling time of $10ms$ (milliseconds). For data real-time visualization and saving, we use the LabView software [65].

A measurement campaign was conducted with constant input velocity with three different friction levels by varying the load on the contact pin, namely: *nominal* (no contact), *moderate* (5 N), and *severe* (10 N) conditions.

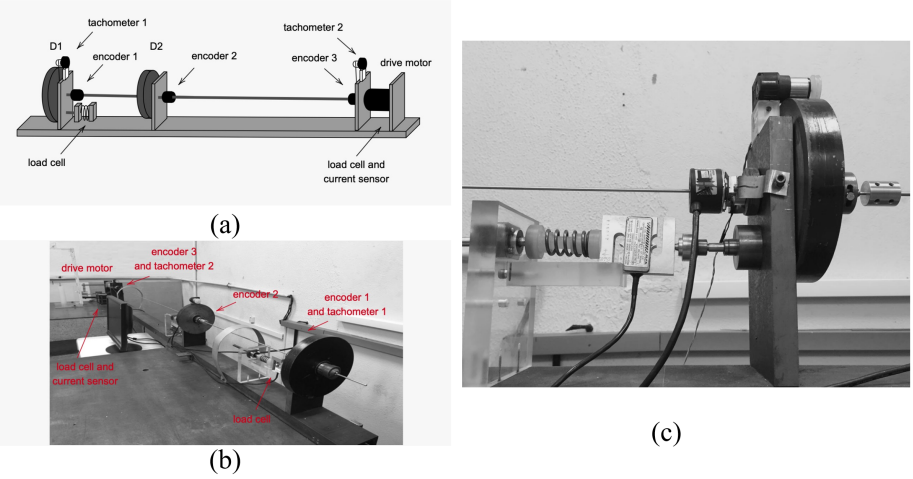


Figure 4.1: Test rig schematic description (a) and actual picture (b). The variable friction is emulated with a force-controlled pin that adds resistive torque to the second disk, which is detailed in (c).

The velocity-time history (RPM-sec) in the three friction levels is illustrated in Figure 4.2. The goal hereafter is to construct an unsupervised data-driven modeling workflow capable of distinguishing such conditions and building an online monitoring tool.

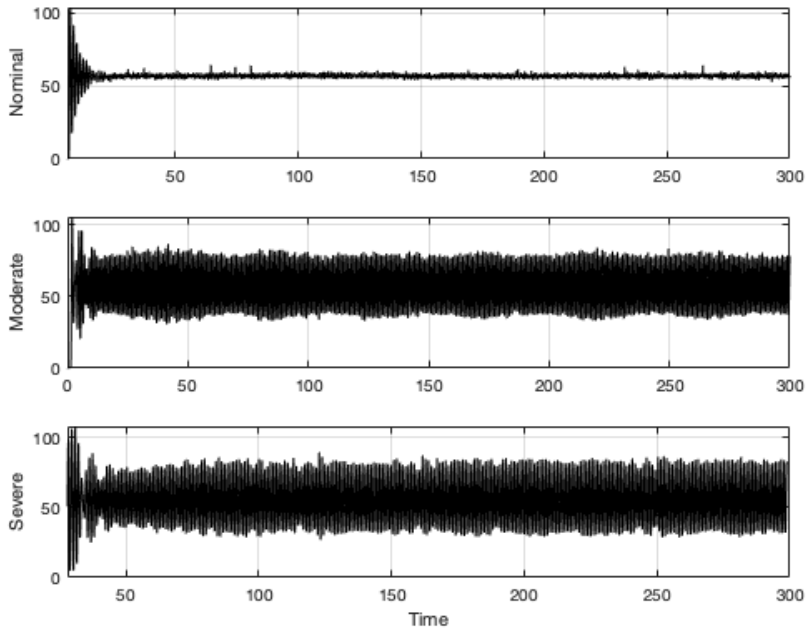


Figure 4.2: Velocity-time history in three friction levels

4.1.1 Embedded Computing Platform

In the scope of the present research, the Nvidia Jetson Nano[1] was used, it is illustrated in Fig 4.3. It is a relatively cheap embedded system-on-module (SoM), including an integrated 128-core Maxwell GPU, quad-core ARM A57 64-bit CPU, 4GB LPDDR4 memory, and support for MIPI CSI-2 and PCIe Gen2 high-speed I/O.

Figure 4.3: Nvidia Jetson Nano [66]

The same allows the user to develop distributed machine learning applications. Furthermore, such a platform is handy for vibration monitoring of rotating machines, given the sampling time required, which is usually unavailable in traditional SCADA systems [67].

4.2 Results and Analysis

The overall method for embedded anomaly detection is described in Figure 4.4. The signals are windowed using a receding-horizon strategy to generate an input-output tuple database, then fed to the feature extraction phase using PCA or autoencoders. The models are then created using PCA and shallow learning models, while the autoencoder is plugged into a FFANN classifier using transfer learning. Finally, the models are embedded to assess their performance for real-time inference.

4.2.1 Dimensionality Reduction

An autoencoder architecture was used with (50,25,3,25,50) hidden neurons architecture. The bottleneck with three neurons represents the reduced space. A plot of the bottleneck representation and the reconstruction of the

[1] `https://developer.nvidia.com/embedded/jetson-nano-developer-kit`

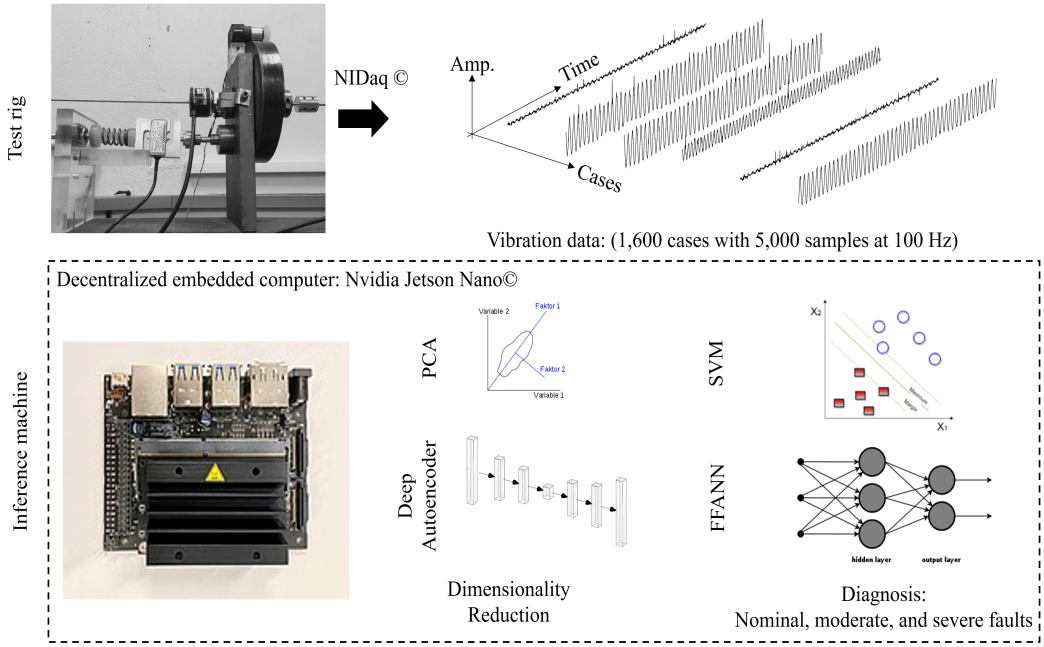


Figure 4.4: Description of the experimental setup data and the embedded vibration monitoring tool.

encoder is illustrated in Fig. 4.5. One can see that there is a reconstruction error, and there is room to improve the autoencoder.

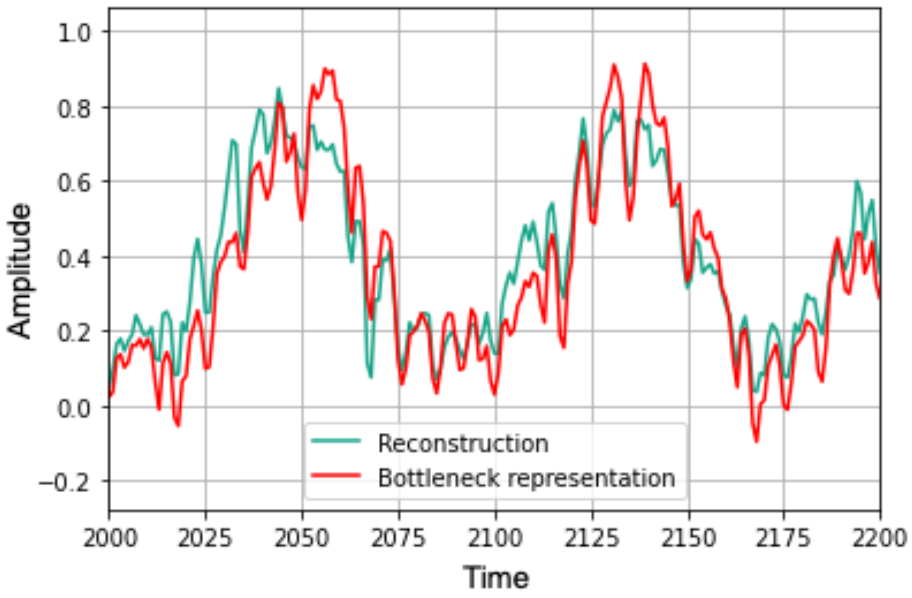


Figure 4.5: Comparison plot of the training set and the reconstruction of the encoder in the final output of the network

The PCA was set to retain 95% of the variance, which resulted in only

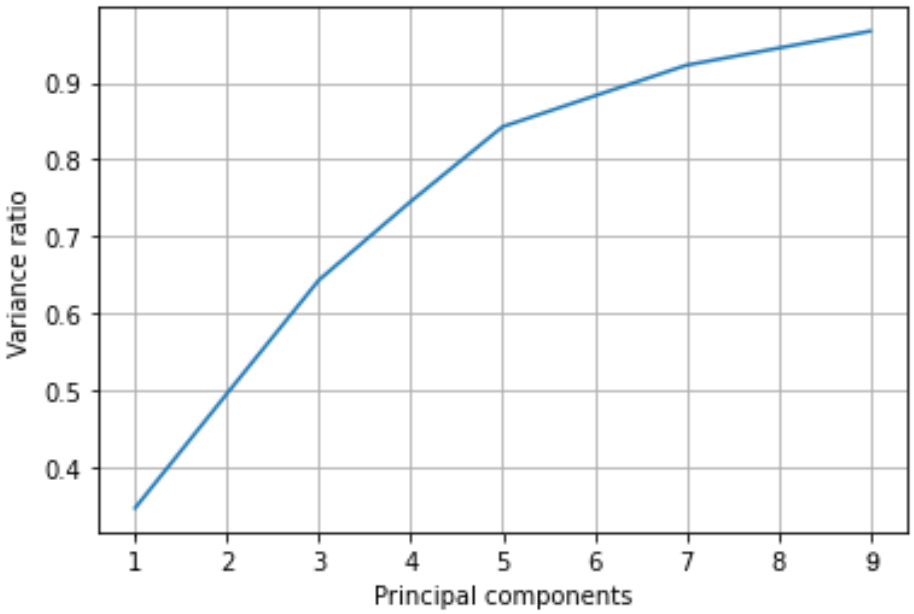
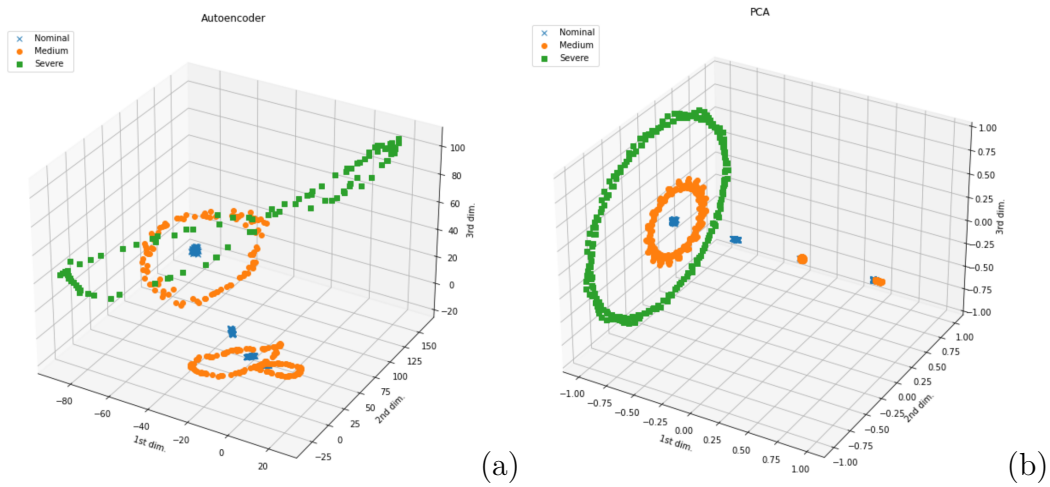


Figure 4.6: Explained variance ratio vs. total principal components.

nine principal components out of 5,000 dimensions. The explained variance is illustrated in Fig.4.6.

Both autoencoder and PCA results are shown in Fig. 4.7a and Fig. 4.7b, which displays great results in terms of class separability. The nominal friction level data, illustrated in blue, is concentrated together in little blocks of data that can be distinguished from the medium and severe friction level data that are more sparsed in the plane but still concentrated together.



(a) (b)

Figure 4.7: Unsupervised learning (a) results of the unsupervised dimensionality reduction using autoencoder and (b) PCA

The architecture of the autoencoder is given in Fig. 4.8.

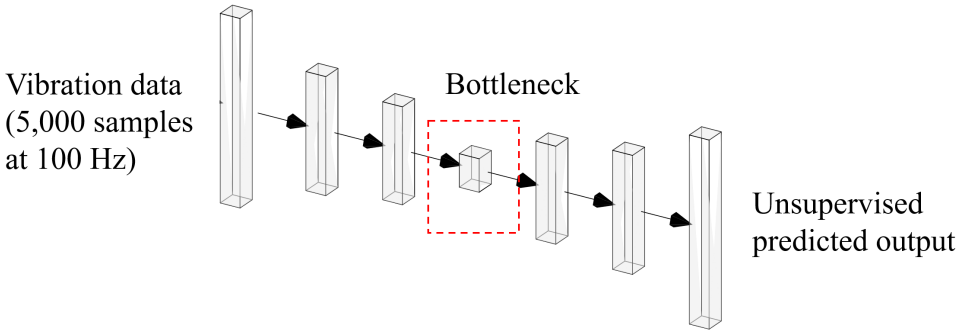


Figure 4.8: Autoencoder architecture depicted.

4.2.2 Model Performance Analysis

The results of the models are depicted in Figure 4.9 in 50% of holdout data. The hyperparameters used in this case were the default parameters, listed in Table 4.1. The model is compiled using the Adam optimizer with 0.001 of learning rate.

Table 4.1: Model hyperparameters

LGR	
C values	1.0
Solver	"rbf"
SVM	
C values	1.0
Kernel type	“rbf”
Degree	3
Gamma	“scale”
ANN	
Unit	10 and 3
Activation Function	"Elu" and "Softmax"

The outputs of the autoencoder are attached to a feedforward neural network using transfer learning and retrained using supervised learning. A sequential model was used using Keras API from Tensorflow library to add a trainable classifier on top. The activation function is used to introduce a nonlinearity on each node of the neural network. For this particular case, the Exponential Linear Unit (Elu) and Softmax were used. Softmax converts a vector of value to a probability distribution. The model’s output shape vary

depending on the layer. There are 3 dense layers of output (None, 10) and one dense layer of output (None, 3), same as the encoder.

The increase of friction causes the increase of amplitudes of vibration, justifying the use of the velocity signal time series to distinguish the system's friction level. One may observe a clear difference between the nominal, moderate, and severe friction conditions in both results of unsupervised dimensionality reduction displayed in Fig. 4.7a and b. This difference is better recognized in the PCA results. Figure 4.9 confirms the better performance of shallow models for the data collected and used in this analysis. The SVM model outperforms other models with respect to accuracy, while the linear model is the fastest. The autoencoder performs poorly when compared to SVM.

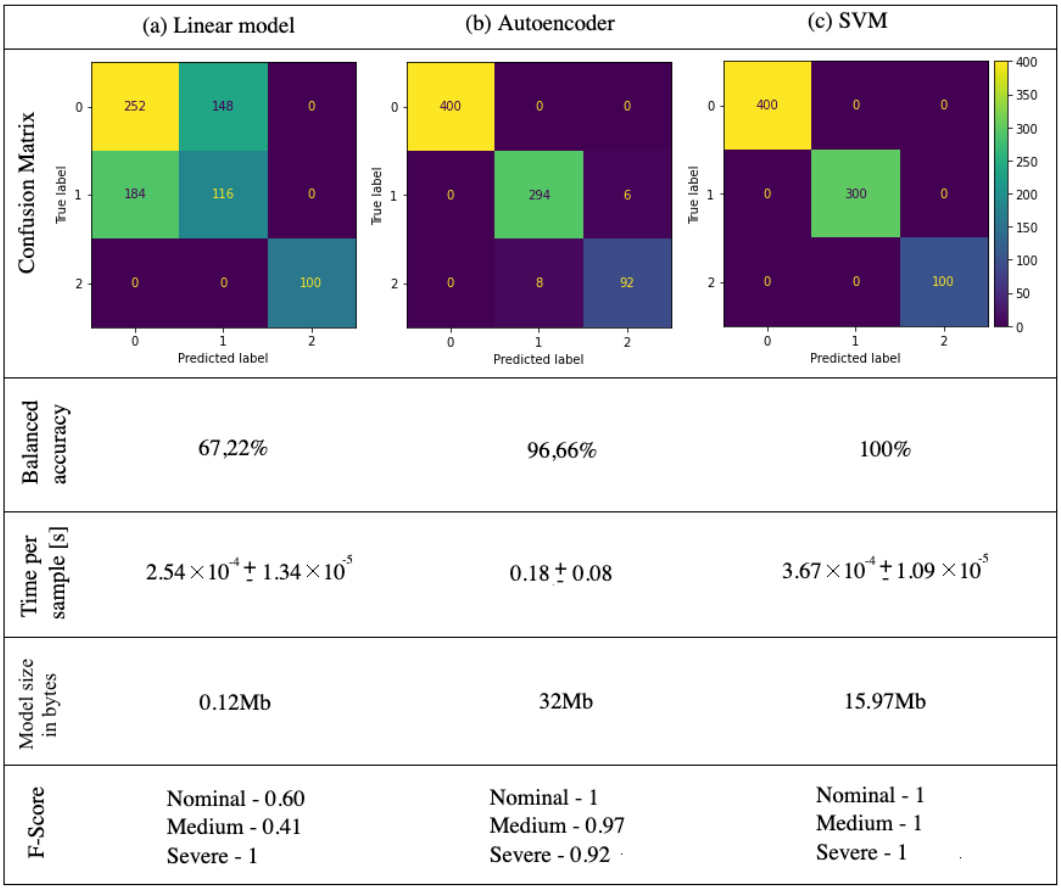


Figure 4.9: Evaluation metrics in holdout in terms of confusion matrices, balanced accuracy, inference time (in seconds) and byte size in Mb for the linear model using the embedded platform (a), autoencoder (b), and SVM (c)

As seen in Figure 4.9, the linear model is used as a baseline and performs poorly compared to the autoencoder and SVM, which present great accuracy. However, the computing time is much shorter for the SVM, which is essential for computational resource optimization. When it comes to the size of the

models, the linear model being the simplest one, has smaller size compared to the others. The encoder associated with the FFANN is the biggest one, having 32Mb in size, since it is the more robust of the models.

Nonetheless, rotating machines may be subjected to a range of friction values and not specific levels, increasing the complexity of fault detection. Deep models may better address this complexity. Additionally, we have shown that the models may be easily embedded on a small platform suitable for edge computing, which provides real-time performance for data processing (see Fig. 4.9 for execution time-per-sample).

Figure 4.9 also lists the F-score values of the classification, calculated from the precision and recall using the Scikit Learn metrics library. The best case scenario happens when F-Score is equal to 1 and worst case scenario when F-Score is equal to 0. Since this is a multi-class classification, the F-scores represents the average of the scores of each class. This values confirms the god precision of the SVM classification of the data, being simpler and more effective.

4.3 Chapter Remarks

The present work has shown an efficient implementation of an embedded system of machine learning models to diagnose rotating machinery using vibration data at high-frequency rates. Shallow models have shown better results in running time, memory usage, and accuracy. Furthermore, feature reduction running online with model inference shows that a cheap GPU-based platform can be used to enable decentralized diagnosis running at high-frequency rates, which is appropriated in the edge computing paradigm.

The feature extraction process made by PCA and autoencoders was effective in separating nominal and abnormal friction states, indicating that unsupervised learning would be applicable for the task at hand [68]. As labeled data is expensive to obtain, such a procedure certainly adds practical appeal for applications if available. Moreover, the autoencoder results show that it might be a better solution for cases involving more complex failure modes.

5
Conclusions

5.1
Final considerations

This research aimed to identify damage in two structures using different approaches from the comprehensive range of machine learning options. The used models were based on two structures, a static one represented in the three-story building of the Los Alamos Laboratory; and a rotational one capable of reproducing torsional behavior like the stick-slip phenomenon.

As data volumes grow, finding optimized models through more efficient data modeling pipelines is important as smaller models lead to better embedded solutions, achieving good results in efficiency, computational effort, and performance.

In the first case, we proved that a correlation between the size of the models and the accuracy exists and can provide better behavior even when the models' size is reduced using dimensionality reduction models like PCA, for example. After the reduction of the model size and a hundred of random validation experiments, the results of the reduced models proved to have good accuracy in the classification and description of the model, proving that smaller versions of the original model can perform well. Furthermore, results show that the closer to the damage source the data acquisitor is, the better the accuracy results. Using more than one data source can increase these results even if it is far from the damage source.

In the second case, the identification between normal and abnormal states for the rotating machinery using vibration data at a high-frequency rate was conducted by implementing a shallow and deep embedded system of machine learning models. Results show that shallow models are better when considering running time and accuracy. The feature extraction using PCA, autoencoder and unsupervised learning for the classification step was very successful. Proving that autoencoders can be a good solution for more complex failure models.

5.2 Future work

As a continuation of this work, it is intended to extract the nonlinear parameters of this same system to understand the behaviour of the results related to the size in bytes of the models and the accuracy of the results. Also test different hyperparameters optimization methods and in different classification applications.

Future work also focuses on real-time implementation of hardware of optimized intelligent classifiers using Field Programmable Gate Arrays (FPGAs) as they have a parallel architecture which makes them suitable for machine learning applications, quantization of models with a focus on embedded systems and scalability to lower cost hardware.

References

[1] C. Farrar and K. Worden. *Structural Health Monitoring: A Machine Learning Perspective*. Chichester (UK): John Wiley Sons Ltd., 2012.

[2] Mo. Azimi, A. D. Eslamlou, and G. Pekcan. "Data-Driven Structural Health Monitoring and Damage Detection through Deep Learning: State-of-the-Art Review." In: *Sensors* 20.10 (2020). ISSN: 1424-8220.

[3] *CBS Minesotta: Friday Marks 7 Years Since I-35W Bridge Collapse*. 2014. URL: `https://minnesota.cbslocal.com/2014/08/01/friday-marks-7-years-since-i-35w-bridge-collapse/` (visited on 04/13/2021).

[4] C. R. Farrar, N. A. Lieven, and M. Bement. "Damage Prognosis: For Aerospace, Civil and Mechanical Systems." In: Chichester (UK): John Wiley Sons Ltd., 2005.

[5] L. C. R. Morenghi. *Proposta de um Sistema Integrado de Monitoramento para Manutenção*. M. SC. DIssertation 125p. Escola de Engenharia de São Carlos da Universidade de São Paulo, 2005.

[6] A. Brasiliano. *Identificação de Sistemas e Atualização de Modelos Numéricos com Vistas à Avaliação da Integridade Estrutural*. PHD Dissertation 222p. Universidade de Brasília, 2005.

[7] L. Bornn, C. Farrar, and K. Farinholt. "Structural health monitoring with autoregressive support vector machines." In: *Journal of Vibration and Acoustic* 131 (2009), pp. 0210041–0210049.

[8] E. J. F. Figueiredo. *Damage Identification in Civil Engineering Infrastructure under Operational and Environmental Conditions*. PHD Dissertation 226p. University of Porto, 2010.

[9] E. Figueiredo et al. "Structural health monitoring algorithm comparisons using standard data sets." In: *Los Alamos National Laboratory Report: LA-14393* (2009).

[10] J. LI et al. "Damage identification in civil engineering structures utilizing PCA-compressed residual frequency response functions and neural network ensembles." In: *Struct. Control Health Monit* 18(2) (2011), pp. 207–226.

[11] T. Nguyen, T.H.T. Chan, and D.P. Thambiratnam. "Controlled Monte Carlo data generation for statistical damage identification employing Mahalanobis squared distance." In: *Structural Health Monitoring* 13(4) (2014), pp. 461–472.

[12] A. Santos et al. "Machine learning algorithms for damage detection: Kernel-based approaches." In: *Journal of Sound and Vibration* 363 (2016), pp. 584–599.

[13] G. Gui et al. "Data-Driven Support Vector Machine with Optimization Techniques for Structural Health Monitoring and Damage Detection." In: *KSCE Journal of Civil Engineering* 21(2) (2017), pp. 523–534.

[14] H. Pan, Z Lin, and G. Gui. "Enabling Damage Identification of Structures Using Time Series–Based Feature Extraction Algorithms." In: *J. Aerosp. Eng.* 32(3) (2019), pp. 04019014-1 –15.

[15] B Zhang, K-M. Hong, and Y. C. Shin. "Deep-learning-based porosity monitoring of laser welding process." In: *Manufacturing Letters* 23 (2020), pp. 62–66.

[16] Z. Ye and J. Yu. "AKSNet: A novel convolutional neural network with adaptive kernel width and sparse regularization for machinery fault diagnosis." In: *Journal of Manufacturing Systems* 59 (2021), pp. 467–480.

[17] D. F. Hesser and B. Markert. "Tool wear monitoring of a retrofitted CNC milling machine using artificial neural networks." In: *Manufacturing Letters* 19 (2019), pp. 1–4. ISSN: 22138463.

[18] S. Han et al. "Classification and regression models of audio and vibration signals for machine state monitoring in precision machining systems." In: *Journal of Manufacturing Systems* 61 (2021), pp. 45–53.

[19] O. Avci et al. "A review of vibration-based damage detection in civil structures: From traditional methods to machine learning and deep learning applications." In: *Mechanical Systems and Signal Processing* 147 (2021), pp. 107077–107077.

[20] M. Mozaffar et al. "Data-driven prediction of the high-dimensional thermal history in directed energy deposition processes via recurrent neural networks." In: *Manufacturing Letters* 18 (2018), pp. 35–39. ISSN: 22138463.

[21] J. Francis and L. Bian. "Deep Learning for Distortion Prediction in Laser-Based Additive Manufacturing using Big Data." In: *Manufacturing Letters* 20 (2019), pp. 10–14.

[22] O. Avci et al. “A review of vibration-based damage detection in civil structures: From traditional methods to Machine Learning and Deep Learning applications.” In: *Mechanical Systems and Signal Processing* 147 (2021), p. 107077.

[23] E. Moura et al. “Evaluation of principal component analysis and neural network performance for bearing fault diagnosis from vibration signal processed by RS and DF analyses.” In: *Mechanical Systems and Signal Processing - MECH SYST SIGNAL PROCESS* 25 (2011), pp. 1765–1772.

[24] O. Abdeljaber et al. “Real-time vibration-based structural damage detection using one-dimensional convolutional neural networks.” In: *Journal of Sound and Vibration* 388 (2017), pp. 154–170. ISSN: 10958568.

[25] S. Mittal. “A Survey on optimized implementation of deep learning models on the NVIDIA Jetson platform.” In: *Journal of Systems Architecture* 97 (2019), pp. 428–442.

[26] R. Giubilato et al. “An evaluation of ROS-compatible stereo visual SLAM methods on a nVidia Jetson TX2.” In: *Measurement* 140 (2019), pp. 161–170.

[27] Z. Pan et al. “A two-stage method based on extreme learning machine for predicting the remaining useful life of rolling-element bearings.” In: *Mechanical Systems and Signal Processing* 144 (2020), p. 106899.

[28] A. S. P. Aguiar et al. “Vineyard trunk detection using deep learning – An experimental device benchmark.” In: *Computers and Electronics in Agriculture* 175 (2020), p. 105535.

[29] A. Verma et al. “Edge-cloud computing performance benchmarking for IoT based machinery vibration monitoring.” In: *Manufacturing Letters* 27 (2021), pp. 39–41. ISSN: 22138463.

[30] B. C. Cayres. *Numerical and experimental analysis of nonlinear torsional dynamics of a drilling system*. M.Sc. Dissertation 88p. Pontifícia Universidade Católica do Rio de Janeiro, 2013.

[31] J.L. Klein. *Statistical Visions in Time: A History of Time Series Analysis*. Cambridge, U.K.: Cambridge University Press, 1997, pp. 1662–1938.

[32] E. Figueiredo et al. “Influence of the Autoregressive Model Order on Damage Detection.” In: *Computer-Aided Civil and Infrastructure Engineering* 26 (2011), pp. 225–238.

[33] I.T. Jolliffe and J. Cadima. “Principal component analysis: a review and recent developments.” In: *Phil. Trans. R. Soc. A* 374 (2016), p. 20150202.

[34] H. Hotelling. "Analysis of a complex of statistical variables into principal components." In: *Journal of Educational Psychology* 24(6) (1933), pp. 417–441.

[35] S.L. Brunton and J.N. Kutz. *Data Driven Science Engineering: Machine Learning, Dynamical Systems, and Control*. London: Cambridge University Press, 2019.

[36] B.E. Boser, I. G. Guyon, and V.N. Vapnik. "A Training Algorithm for Optimal Margin Classifiers." In: *Annual Workshop on Computational Learning* 5 (1992), pp. 144–152.

[37] C-C. Chang and C-J. Lin. "LIBSVM: A Library for Support Vector Machines." In: *ACM Transactions on Intelligent Systems and Technology* 2(3) (2011), pp. 1–27.

[38] J.S. Cramer. "The Origins of Logistic Regression." In: *Tinberger Institute Discussion Paper* 119(4) (2002), pp. 167–178.

[39] M.P. LaValley. "Logistic Regression." In: *Statistical Primer for Cardiovascular Research* 117 (2008), pp. 2395–2399.

[40] S. Sperandei. "Understanding logistic regression analysis." In: *Biochemia Medica* 24(1) (2014), pp. 12–18.

[41] L. Breiman et al. *Classification and regression trees*. Monterrey, CA: Wadsworth Brooks/Cole Advanced Books Software, 1984.

[42] R. Agrawal. "K-Nearest Neighbor for Uncertain Data." In: *International Journal of Computer Applications* 105(11) (2014), pp. 13–16.

[43] L. Rokach and O. Maimon. *Data Mining and Knowledge Discovery Handbook. Chapter 9 – Decision Trees*. Boston: Springer, 2005.

[44] E. Fix and J. L. Hodges. "Discriminatory Analysis. Nonparametric Discrimination: Consistency Properties." In: *USAF School of Aviation Medicine* (1951).

[45] T.M. Cover and P.M. Hart. "Nearest Neighbor Pattern Classification." In: *IEEE Transactions on Information Theory* 13(1) (1967), pp. 21–27.

[46] L. Breiman. "Random Forests." In: *Machine Learning* 45(1) (2001), pp. 5–31.

[47] A. Géron. *Hands-On Machine Learning with Scikit-Learn and TensorFlow*. USA: O'Reilly Media, Inc., 2017.

[48] S. Bhatnagar, L. Gill, and B. Ghosh. "Drone Image Segmentation Using Machine and Deep Learning for Mapping Raised Bog Vegetation Communities." In: *Remote Sensing* 12 (2020), p. 2602.

[49] F. Rosenblatt. “The perceptron: a probabilistic model for information storage and organization in the brain.” In: *Psychological Review* 65(6) (1958), pp. 386–408.

[50] D. M. D. Sereno. *Automatic Evolution of Deep Autoencoders*. M.Sc. Dissertation 50p. University of Coimbra, 2018.

[51] S. Di Frischia and F. Angelini. “Evaluation of Artificial Neural Networks performances on spectral classification tasks.” In: (June 2019).

[52] T Vauyapri and A. Binbusayyis. “Enhanced deep autoencoder based feature representation learning for intelligent intrusion detection system.” In: *Computers, Materials Continua* 68 (2021), pp. 3271–3288.

[53] M. Z. Sarwar and D. Cantero. “Deep autoencoder architecture for bridge damage assessment using responses from several vehicles.” In: *Engineering Structures* 246 (2021), p. 113064.

[54] F. Roche et al. “Autoencoders for music sound modeling: a comparison of linear, shallow, deep, recurrent and variational models.” In: May 2019.

[55] T. Agrawal. *Hyperparameter Optimization in Machine Learning: Make Your Machine Learning and Deep Learning Models More Efficient*. Apress, 2020.

[56] K. E. S. Pilario, Y. Cao, and M. Shafiee. “A Kernel Design Approach to Improve Kernel Subspace Identification.” In: *IEEE Transactions on Industrial Electronics* PP (2020), pp. 1–1.

[57] D Kingma and J. Ba. “Adam: A Method for Stochastic Optimization.” In: *International Conference on Learning Representations* (2014).

[58] S. Arlot and A. Celisse. “A survey of cross-validation procedures for model selection.” In: *Statistics Surveys* 4.none (), pp. 40–79.

[59] F. Maleki et al. “Machine Learning Algorithm Validation: From Essentials to Advanced Applications and Implications for Regulatory Certification and Deployment.” In: *Neuroimaging Clinics of North America* 30.4 (2020). Machine Learning and Other Artificial Intelligence Applications, pp. 433–445.

[60] R. R. Picard and R. D. Cook. “Cross-Validation of Regression Models.” In: *Journal of the American Statistical Association* 79.387 (1984), pp. 575–583.

[61] A. Lendasse, V. Wertz, and M. Verleysen. "Model Selection with Cross-Validations and Bootstraps – Application to Time Series Prediction with RBFN Models." In: *ICANN/ICONIP 200, LNCS 2714* (2003), pp. 573–580.

[62] Q-S. Xu and Y-Z. Liang. "Monte Carlo cross validation." In: *Chemometrics and Intelligent Laboratory Systems* 56.1 (2001), pp. 1–11. ISSN: 0169-7439.

[63] R-E. Fan et al. "LIBLINEAR: A Library for Large Linear Classification." In: *Journal of Machine Learning Research* 9 (2008), pp. 1871–1874.

[64] Bruno C. Cayres et al. "Analysis of a Drill-String Experimental Set-Up with Dry Friction-Induced Torsional Vibration." In: *Proceedings of the 10th International Conference on Rotor Dynamics – IFToMM*. Ed. by Katia Lucchesi Cavalca and Hans Ingo Weber. Cham: Springer International Publishing, 2019, pp. 456–469.

[65] Ingrid Pires et al. "Torsional Friction-Induced Vibrations in Slender Rotating Structures." In: *Advances in Mechanism and Machine Science*. Ed. by Tadeusz Uhl. Cham: Springer International Publishing, 2019, pp. 3449–3458.

[66] *Jetson Nano Developer Kit*. 2021. URL: `https://developer.nvidia.com/embedded/jetson-nano-developer-kit` (visited on 04/21/2021).

[67] E. Gonzalez et al. "Using high-frequency SCADA data for wind turbine performance monitoring: A sensitivity study." In: *Renewable Energy* 131 (2019), pp. 841–853.

[68] L. Ruff et al. "A Unifying Review of Deep and Shallow Anomaly Detection." In: *Proceedings of the IEEE* 109.5 (2021), pp. 756–795.