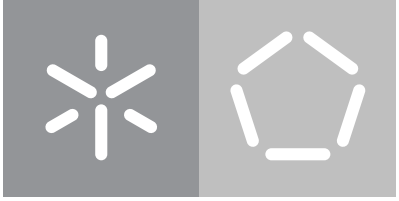
Universidade do Minho
Escola de Engenharia

Bruno Manuel Macedo Nascimento

Detection and classification of small impacts on vehicles based on deep learning algorithms

July, 2022



Universidade do Minho
Escola de Engenharia

Bruno Manuel Macedo Nascimento

Detection and classification of small impacts on vehicles based on deep learning algorithms

Master's Dissertation
Integrated Master's in Informatics Engineering.

Thesis supervised by
João Miguel Lobo Fernandes
André Leite Ferreira

July, 2022

STATEMENT OF INTEGRITY

I hereby declare having conducted this academic work with integrity. I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration.

I further declare that I have fully acknowledged the Code of Ethical Conduct of the Universidade do Minho.

_____, _____

(Location) (Date)

_____

(Bruno Manuel Macedo Nascimento)

Acknowledgements

I would like to express my gratitude and appreciation for everyone whose guidance, support and encouragement has been invaluable throughout this study. I also wish to thank the team at BOSCH Car Multimedia S.A. who have been a great source of support.

Abstract

This thesis explores the detection of impacts that cause damage based on data retrieved by an accelerometer placed inside a vehicle and subsequently classified by deep learning algorithms. The real world application of this work inserts itself in the car sharing market, by providing an automated service that allows constant monitoring on the vehicle status.

The proposed solution was set as an alternative to the current machine learning algorithms in use. Previous research showed that deep learning algorithms are achieving better performance results when compared to non deep learning algorithms.

We use data retrieved from two types of events: Normal driving and damage causing situations to test if the models are capable of generalising damage events. The approach to achieve this objective consisted in exploring and testing different algorithms: Multi Layer Perceptron (MLP), Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN).

Results revealed promising performance, with the MLP reaching a 82% true positive rate. Despite not matching the result obtained by the current non deep learning algorithm allows us to assess that deep learning is a strong alternative in the long term as more data is collected.

Keywords: Impact Detection, Artificial Intelligence, Deep Learning, Neural Network, Signal Processing

Resumo

O principal objectivo desta tese foi a exploração e detecção de impactos que causam danos com base em dados recolhidos por um acelerómetro colocado no interior um veículo e posteriormente classificados por algoritmos de *deep learning*. A aplicação deste trabalho no mundo real insere-se no mercado de partilha de veículos, ao fornecer um serviço automático que permite uma monitorização constante do estado do veículo.

A solução proposta foi definida como uma alternativa aos actuais algoritmos de *machine learning* em uso. A revisão de literatura revelou que algoritmos de *deep learning* estão a alcançar melhores resultados de desempenho quando comparados com algoritmos de *machine learning*.

Utilizamos dados recolhidos de dois tipos de eventos: Condução normal e situações que causam dano e testar se os modelos são capazes de generalizar os eventos de danos. A abordagem para alcançar este objectivo consistiu em explorar e testar diferentes algoritmos: MLP, CNN e RNN.

Os resultados revelaram um desempenho promissor, com a MLP a atingir uma taxa de 82% de verdadeiros positivos. Apesar de não corresponder ao melhor resultado obtido pelo actual algoritmo de *machine learning* em uso permite-nos avaliar que *deep learning* é uma forte alternativa a longo prazo à medida que mais dados forem recolhidos.

Palavras-chave: Detecção de Impactos, Inteligência Artificial, Deep Learning, Redes Neuronais, Processamento Síntese de Sinal

Contents

List of Figures

List of Tables

Acronyms

ABS	Anti-Lock Brake System
AI	Artificial Intelligence
ANN	Artificial Neural Network
BLSTM	Bidirectional Long Short Term Memory
CNN	Convolutional Neural Network
DAN	Deep Autoencoder Network
DAR	Deep Audio Representation
DB	Driving Behaviour
DCNN	Deep Convolutional Neural Network
DFT	Discrete Fourier Transform
eCall	Emergency Call
EDA	Exploratory Data Analysis
ELM	Extreme Learning Machine
EMS	Emergency Medical Services
ESP	Electronic Stability Control
FCN-LSTM	Fully Convolutional Long Short Term Memory
FFT	Fast Fourier Transform
FN	False Negative
FP	False Positive
GPS	Global Positioning System
GRU	Gated Recurrent Unit
HDF	Hierarchical Data Format

ID	Impact Detection
IMU	Inertial Measure Unit
IS	Inertial Sensor
JSON	JavaScript Object Notation
KNN	K-Nearest Neighbours
LDA	Linear Descriminant Analysis
LSTM	Long Short Term Memory
MCC	Mathews Correlation Coefficient
MEMS	Micro-Electro-Mechanical Systems
MFCC	Mel Frequency Cepstral Coefficients
ML	Machine Learning
MLP	Multi Layer Perceptron
MSDF	Multi Sensor Data Fusion
NN	Neural Network
OEMs	Original Equipment Manufacturers
PCA	Principal Component Analysis
PRNN	Persistent Recurrent Neural Network
RF	Random Forest
RNN	Recurrent Neural Network
SFFS	Sequential Forward Feature Selection
SOM	Self-organizing Map
SVM	Support Vector Machine
TN	True Negative
TP	True Positive

Chapter 1

Introduction

1.1 Context

With the continuous research and development in artificial intelligence, machines are able to learn faster and more efficiently through machine learning and deep learning techniques. Deep learning is currently revolutionizing the world in several areas by providing new solutions to existing problems. The challenge that this thesis proposes to address is expected to benefit from this technology.

Car sharing is an alternative to owning a vehicle by allowing for a fast and accessible approach to move around within a city. The vehicle can be picked and dropped at any place and any hour without human interaction.

Improper actions may cause damage that can go unnoticed and without the human inspection done after every use the vehicle quality and operation could be at risk. In order to solve this problem an approach is based on the data retrieved by an accelerometer placed inside the vehicle. This data is applied to a deep learning algorithm to classify whether an event that passes a certain threshold set to delineate impact with damage or not is the main focus of this thesis.

This thesis was developed at BOSCH CAR MULTIMEDIA S.A., located in Braga, Portugal, as an internship collaboration between BOSCH and University of Minho. The company is currently developing a product for a car group, to later deploy in a fleet of vehicles across the globe under a mobility solution for city centers as a car sharing business available for the general public to use as an on demand service. With intent of replacing the actual business of car renting with a friendly, quicker and efficient system using current technologies and future ones.

For software development, BOSCH uses Scrum agile methodology. Each phase is scheduled into a sprint for an easier planning and control, leading to a fulfillment of objectives and improvement over time thanks to a iterative process resulting in a rapid implementation and deployment of a product.

1.2 Motivation

Implement a system that detects damage to a vehicle from impact with other vehicles, structures or objects. This monitoring allows a car sharing fleet operator to classify the condition of a vehicle after it has been rented by a customer allowing real-time fleet control without the need to view the car in person, thus avoiding the unavailability of the vehicle for damage analysis.

1.3 Objectives

The main objective is the development of a deep learning model capable of detecting and classifying damages caused by an impact onto a vehicle, and benchmark the performance against a Random Forest (RF) based approach currently in use. This is requested by Bosch as they plan to use deep learning for future projects and leave traditional Machine Learning (ML) algorithms.

The current score achieved by this RF approach is a 90% true positive rate, being this value the benchmark to compete against.

Another objective of this thesis is to cycle through an Artificial Intelligence (AI) development life cycle starting with a planning phase and ending in the production phase.

1.4 Document Outline

This document is organized into the following chapters:

- **Introduction** : Presents the context, motivation and objectives defined in the pipeline.
- **State Of The Art**: Related search of relevant concepts and a statistical analysis made on the topics research.
  - Machine Learning: Study and research of techniques and detailed explanation on deep learning.
  - Impact detection: Two approaches studied, the first one being based on data collected by accelerometers, gyroscopes and Inertial Measure Unit (IMU) and techniques used to classify events. The second one is based on audio data collected by microphones with the study of techniques used to classify the events based on the audio signal.
  - Driving Behaviour: Relevant studies and concepts related to deep learning techniques were researched to adjust this topic to impact detection.
- **Problem statement and proposed solution**: Definition of the problem to be solved from a technical point of view and the development methodology used.

- **Data collection for detection of anomalies in a vehicle's cockpit project**: Presents the planning and execution of data collection and analysis made for the detection of anomalies inside a vehicle's cockpit with data retrieved by audio and gas sensors.
- **Accelerometer sensor data understanding and analysis approach for damage detection**: Presents data analysis and understanding of damage and non-damage events based on data retrieved by an accelerometer.
- **Modelling and Evaluation**: Development and evaluation of Neural Network (NN) models with the accelerometer data to detect and classify damage caused by impact onto a vehicle.
- **Conclusion and Future Work**: Summary of all the work done of this thesis with discussion on what was achieved and future work to be considered.

1.5 Workplace Methodology

Agile is a methodology where the entire Software Development Life Cycle(SDLC) is sliced in continuous iterations and testing. And this is where SCRUM steps in by acting as an agile framework that lends steps to manage and control the software and product development. SCRUM is the combination of iterative and incremental features to accelerate the speed of development.

As previously said, this thesis was being written under a curricular internship at Bosch Car Multimedia S.A, where the development of software is done under SCRUM methodology. Interns are introduced to the methodology with a brief overview and explanation.The cycle starts with interns entering *sprint 0*, to have a better feeling of what to expect in the course of the internship. Interns are introduced to a team composed by: Scrum Master and Product Owner.

The Scrum workflow consists in the following components:

- Backlogs: Requirements to be made during a timestamp.
- Sprints: Timestamp duration of the Backlog, most frequent time is 14 days.
- Scrum meetings: Everyday a 15 minutes meeting to know the how the process is going, what is being made, what is not being made and why.
- Demos: Delivery and evaluation of software made until that phase.

Chapter 2

State Of The Art

The development of this thesis focuses primarily on the usage of ML techniques to attempt to solve the perceived problem of small impact detection and classification within a vehicle. This section presents a brief introduction to the following topics:

- Machine Learning
  - Feature engineering
  - Models
  - Deep learning
- Impact detection
  - Audio signal processing
  - Inertial sensors
- Driving behaviour

Every topic detailed in the following subsections is an important basis to understand what is the present view regarding research and analysis within impact detection and classification. In each topic, an overview on what is being done is described with references backing up the search.

2.1 Machine Learning

ML is the study and development of algorithms that allow a computer to learn. Learning is defined as the ability to recognize patterns in a set of data that can then be used to make predictions, such as classifying similar but never seen before data instances or predicting the results of a certain event (Mitchell,

1997). These algorithms often work by using statistical techniques to estimate the probability distribution of the data (Goodfellow et al., 2016).

ML approaches are usually divided between supervised learning and unsupervised learning. In supervised learning, evaluation and correction are made by the supplier to direct the algorithm towards what exactly is meant to be learned. This indicator frequently takes the shape of labels which are the value that the algorithm must learn to predict for each of the training examples. On the other hand, in the case of unsupervised learning no such evaluation and correction exists, and therefore the learning algorithm merely captures patterns in the data that are meant to be relevant according to a variety of metrics including for instance how often pattern frequency shows up in the dataset. It is then up to the supplier to attempt to make the most out of these patterns and apply them where they are most useful. There are also other important sub-groups that are important enough to be worth mentioning for a good overview of the field such as reinforcement learning and semi-supervised learning. Reinforcement learning unlike other approaches is applied to dynamic environments and provided with a reward signal that tells the algorithm how well it is performing. The algorithm must then learn how to act in order to maximize its future reward. Semi-supervised learning is similar to supervised learning but only a subset of the dataset is labeled, the rest is unlabeled (Goodfellow et al., 2016).

One of the main problems that ML tries to tackle is the problem of generalization, or in other words, whether the patterns learned from the data apply to the new examples it will possibly see in the future. If an algorithm does not generalize well, and the learned patterns only apply to the data already seen then it can not be said that it actually learned anything particularly useful. In order to properly evaluate an algorithm capability to generalize it is frequent to divide the dataset between a training set and a test set. The learning algorithm tries to create an algorithm as accurate as possible based on the data on the training set but without any information about the test set. Then the algorithm is used on the test set to get an idea how well it behaves on previously unseen data but the algorithm itself is not changed during this process, in other words, it does not try to learn from the test set. Two metrics of system performance come out of this training paradigm, training error and test error. High training error, also know as underfitting, shows that the learning algorithm is missing the relevant patterns in the data. This means that the learning algorithm just is not working, it is not contemplating all cases in the test set, incorrectly labelling them and it might be necessary to fundamentally change it somehow, though what the possible changes available are depends on what algorithm is being used. While what truly matters in ML is the test error, or how well the algorithm generalizes, the test error is at best as low as the training error so minimizing the latter can be just as important as minimizing the difference between the two. When this difference is high, when the test error is much higher than the training error, it is said the algorithm has overfitted. This means that the learning algorithm has learned the details of each specific training example rather than the overall patterns that are common to the whole dataset. This severely limits the algorithm's capability to generalize which leads to an inflated test error. Usually the most straightforward way to fix this is getting more training data, unfortunately, this is not always viable. Another common solution is early stopping, where, to stop the algorithm from overfitting, the training process is stopped when the test error starts decreasing (Domingos,

2012; Goodfellow et al., 2016).

2.1.1 Feature Engineering

In ML, choosing features to run the algorithm is one of the most important tasks. This selection affects the performance as it may differentiate a good algorithm from a bad algorithm. The process of feature engineering encompasses the processes of selecting and transforming variables from the input data onto the predictive algorithm being implemented.

After analysing the input data required features should be highlighted, and if needed, additional features should be created to expand useful information. There is also the manipulation of data, specially transforming a feature to improve its performance in the predictive algorithm, it can either be dealing with the data type, normalizing its value for easier perception.

After all these steps a selection of features to be supplied to the algorithm should take place, according to their importance to the desired learning.

2.1.2 Machine Learning Models

As previously mentioned ML is composed of different approaches, supervised and unsupervised learning.

2.1.2.1 Supervised Learning

There are two types of supervised learning, classification and regression.

In regression, a prediction is made to forecast the outcome of a sample when the output variable is in the form of a real value, i.e., *money*, *height*, *weight* ...

Regression is used to predict a continuous target.

The following regression algorithms are used to predict values.

- Linear regression;
- Polynomial regression;
- Exponential regression;
- Logistic regression;
- Logarithmic regression.

In classification, a prediction is made to forecast the outcome of a sample when the output variable is in the form of a category, i.e., *red* or *blue*, *cat* or *dog*, among others.

Classification is used to predict discrete targets.

The following classification algorithms are used to predict items or classes.

- K-Nearest Neighbours (KNN);
- Decision Trees;
- Random Forest (RF);
- Support Vector Machine (SVM);
- Naive Bayes.

2.1.2.2 Unsupervised Learning

There are two types of unsupervised learning, dimensional reduction and clustering.

In dimensional reduction, data is transformed from a high-dimensional space into a low-dimensional, reducing features and retaining only the meaningful. It is commonly used in recommendation systems, where only the user "likes/preferences"count.

The following dimensional reduction algorithms are used to reduce dimensions.

- Principal Component Analysis (PCA);
- Kernel PCA;
- Linear discriminant analysis;
- Autoencoder;
- T-distributed Stochastic Neighbour Embedding.

In clustering, samples are gathered in groups/clusters where objects are similar to each other.

The following clustering algorithms are used to group data points.

- K-Means Clustering;
- Gaussian Mixture Models;
- Hidden Markov Models.

2.1.3 Deep Learning

Artificial Neural Network (ANN) is a deep learning algorithm based on the biological neurons that can be found in our brains. These systems are represented by a series of neurons laid out into layers. For each neuron, a non-linear function is computed based on the inputs given, providing different weights to each one and consequently communicating and passing it to the following layer. At the end of the network the loss value is calculated. The loss value is the difference between the expected and the real output. Afterwards, the back propagation algorithm is used to calculate the gradient of the loss with respect to each of the weights.

This process goes through multiple iterations with the purpose of trying to find the weights that best minimize the loss. Each one of these iterations is called an epoch. To achieve the best performance, the best model receives as input unseen data to determine if the model is capable of generalizing for unseen data. If not, some hyperparameter tuning may be necessary to achieve the desired results.

One of the first examples of ANNs appeared in 1958 under the name perceptron, Rosenblatt (1957), as an implementation of Hebbian learning, a neuroscience theory of how the brain could possibly learn. The deployment of multiple layers of perceptrons creates a MLP network.

2.1.3.1 Convolutional Neural Networks

CNNs are a popular type of deep neural networks that feature purposeful layers that make them specially suited for working with image, video input among others. Composed of convolutional layers, pooling layers and fully connected layers. The convolutional layers make use of the convolution operation to apply a filter to the entire image. One of the advantages of these layers over fully connected layers is the reduced number of parameters. In a fully connected layer the number of parameters is dependent on the size of the input and the size of the layer which in the case of images would mean more than a parameter per pixel of an image. This would translate into longer training times and more chances of overfitting.

Convolutional layers on the other hand are independent of input size. The number of parameters in the layer being dependent only on the size of the filters and the number of filters. As such training is easier and the algorithm gives reduced importance to every single pixel when compared to filter-sized chunks of the picture which makes sense as in practice very rarely does a single pixel carry much weight on the overall information contained in the image.

Pooling layers are used to reduce the size of the representation of the data which means the network uses less parameters, they are also use to detect edges, eyes, nose, corner among others features by using multiple filters. There are many types of pooling layers. However, two stand out from the rest: Max-pooling and average-pooling, in max-pooling a filter is applied throughout the input of the layer returning only the highest value that it captures. While average-pooling returns the average of the values caught by the filter. These layers serve two purposes, continuous reduction of the feature map's spatial size and progressively identifying relevant features. These types of layers in a mostly alternated fashion followed by a few fully-connected layers are the most common CNN algorithm. Each successive convolutional layer is responsible for detecting more complex and abstract feature of the image from the simpler features captured in previous layers (Zeiler et al., 2014).

2.1.3.2 Recurrent Neural Networks

RNN is a type of NN that uses sequential or time series data as input data to process. They are distinguished by their internal state memory, as they take information from prior inputs to influence the current input and output. While traditional neural networks assume that inputs and outputs are independent from each other, the output of a RNN depends on the prior elements within the sequence. While future

events would also be helpful in determining the output of a given sequence, unidirectional RNN cannot account for these events in their predictions.

Storing information contributes to two main problems, vanishing or exploding gradient. Depending on what activation functions are used. It cannot process long sequences if using tanh or ReLU as an activation function.

LSTM algorithm fixes this issue by introducing gates and explicit memory cells, storing previous values and holds it until a forget instruction is called (Hochreiter et al., 1997).

Reference	**Approach**
Hashimoto et al. (2019)	Uses ML algorithms to detect abnormal vibration via piezoelectric sensors, using a combination of CNN and Mel Frequency Cepstral Coefficients (MFCC) and reaches better performance compared to other algorithms.
Marcillo et al. (2020)	Presents an algorithm for predicting crashes with combination of multiple sources of data by merging them. Creating high risk clusters while using KNN algorithm.
Koch et al. (2018)	Presents a developed automated ML algorithm for time series application where from a large number of features extracts the best features to include in a decision tree algorithm.
Domingos (2012, 2017)	Gives a depth analysis and description on machine learning fundamentals.

Table 2.1: Machine Learning Relevant Literature

2.2 Impact Detection

Road traffic accidents are one of the biggest causes of death every year around the globe with the death toll exceeding one million. Road traffic injuries exceed fatalities values by 3 times[1]. Traffic accidents are also the leading cause of death among children 5-14 and young adults 15-29[2]. Efforts to reduce accidents are deployed continuously by governments and Original Equipment Manufacturers (OEMs). With the introduction of new standards and regulations, active and passive safety keeps on getting enhanced in order to decrease the risk of injury or fatality in accidents.

However, other issues are still being faced. Emergency Medical Services (EMS) when called to dispatch to an accident site do not know what to expect, i.e., number of casualties, impact point, airbag deployment, pedestrian hit, among others.

Requirement for a system that could give detailed information about accidents was demanded. The European Union required every new car sold in its territory to be fitted with an Emergency Call (eCall) as standard from April 2018 onwards. Although it gives information about a vehicle impact, it has to be a serious one, where airbags are deployed, making it a threshold for an automated call, or anyone inside the vehicle can press a SOS button to start an eCall. The problem is when there is an impact, where airbags do not deploy and the occupants are unconscious, the system does not activate an eCall, leading to a delay in EMS arrival and consequently increasing the possibility of an injury victim turn into a fatality without treatment as soon as possible.

Improvement in technological sensing paved the way for new detection systems, based either position or audio. Inertial sensors based on Global Positioning System (GPS), accelerometers and gyroscope parameters allow to determine the behaviour being inflicted onto a vehicle. Abundant data is retrieved from sensors simplifying the determination of all factors involved in an impact, allowing the EMS to acknowledge the situation before dispatching to an accident scene.

Audio signals provide extra data points useful for detailed information whereabouts an impact is inflicted, either inside or outside a vehicle. Therefore processing retrieved audio signal is a crucial task in impact detection.

Both subjects: Inertial Sensor (IS) and Audio Signal Processing will be explained with a deep analysis and how the steps in applying them to the real world are processed.

2.2.1 Inertial Sensors

The need for detecting and analysing the universe of motion, vibration and shock opened a path for IS. For each individual application certain requirements are different and the challenge to fulfill those requirements is complex. But due to the fact that they can be applied everywhere makes it one of the most produced electronics currently.

Some areas of application are the following:

[1] WHO - Global status report on road safety 2018
[2] WHO - Global status report on road safety 2018

- Navigation: Position, speed and altitude.
- Automotive: Airbag deployment, Electronic Stability Control (ESP).
- Industrial: Machinery that monitors vibration and wear.
- Consumer products: Orientation sensing, gesture recognition, motion input, image stabilization, fall detection, sports and healthy lifestyle applications.
- Sports: Fall and concussion detection.

No sensor is perfect and as such they often output errors mixed in with the real data when faced with the vast circumstances of the real world. Given this, most applications that rely on sensors must deal with these errors in some way either by ignoring them or filtering them out somehow depending on the situation. Frequently, errors are associated with either systematic errors or random errors caused by incorrect setup or external factors.

The inertial sensor purpose is to measure motion parameters with respect to the inertial space.

There are two types of inertial measuring field sensors:

- For linear inertial displacement: Accelerometers.
- For rotational inertial rate: Angular rate sensors or gyroscopes.

2.2.1.1 Accelerometers

Is a device that measures translational acceleration resulting from the forces acting on it, following Isaac Newton's *second law of motion*[3]. It consists of a small mass connected via spring to an enclosed case. When exposed to acceleration, the mass moves causing movement on the spring attached to it, either contraction or extension depending on what kind of movement was applied to the mass. Most common accelerometers are three single-axis. Each accelerometer has its axis mounted orthogonally to any other mounted in the system. Accelerometers are insensitive to gravitational acceleration and unable to separate the total acceleration from that caused by the presence of gravitational field.

The following accelerometers are currently available:

- Mechanical;
- Surface acoustic waves;
- Piezoelectric;
- Fiber optic;
- Vibrating beam.
- Micro-Electro-Mechanical Systems (MEMS)

[3] F=m.a, a Force **F** acting on a body of mass *m* causes the body to accelerate with respect of inertial space.

2.2.1.2 Gyroscopes

A gyroscope is a device that measures and maintains angular orientation and can measure turn rates by changes to the inertial space its placed on. Inertial properties of a wheel spinning at a high rate of speed are exploited to keep and maintain the direction of its own axis speed in accordance with the principles of conservation of angular movement. Axle orientation changes due external inputs but at a minimum.

The following mechanical gyroscopes are the currently available:

- Dynamically tuned Gyroscope;
- Flex gyro;
- Dual-axis rate transducer.

Since then, other types of gyroscopes have been developed, specially optical and vibrating. Not based on the principles of conservation of the angular movement. Optical gyroscopes are based on the *Sagnac* effect[4] and vibrating gyroscopes are based on the *Coriolis* effect[5].

[4]Causes a phase shift between two waves counter-propagating in a ring interferometer that is rotating, the shift is proportional to the rate of rotation

[5]Induces a coupling between two resonant modes of a mechanical resonator

Reference	Approach
Yee et al. (2018)	Presents Vehicle Collision Detection (VCD) system. An android App which requires data from on-board sensors in conjunction with those available in the smartphone.
Pai et al. (2014), Nath et al. (2018)	Depends on sensor impact to activate GPS module in order to send a message to a provider or an EMS if needed.
Supriya et al. (2020)	Focus on frontal impacts with effective airbag deployment. Accelerometers are located in the front of the vehicle and seat belts. Kalman filters are employed to estimate the measurement values from noisy data as well as for signal level fusion. The Multi Sensor Data Fusion (MSDF) combines two accelerometers in order to reduce crash sensing time when compared to just one accelerometer, improving efficiency and reliabitlity in event detection.
Gontscharov et al. (2014)	Uses MSDF with inclusion of additional data from the vehicle state, closed or open door status, speed, changing fuel level. In the end the calculated cumulative sum validates the impact and classification.
Collin et al. (2019)	Introduces the development and application of MEMS technology, describing both the generated motion principle and the sensor operating principle.
Parviainen et al. (2014), Selmanaj et al. (2017), Cismas et al. (2017)	Target crash detection in motorcycles due to their complex physics, Uses IMU in the biker body.

Table 2.2: Inertial Sensor Relevant Literature

2.2.2 Audio Signal Processing

Audio has always been a crucial part in human sensing. The ability to process audio and abstract it from the environment is one of the most complex functions that humans can do. Trying to implement an auditory scene analysis on a computing level has always been an desired ambition to provide a better understanding and comprehension in speech recognition and audio detection. To achieve this assignment, the audio must be processed into signal models for better detection and classification.

Audio is composed of physical features which are low-level signals that combine both temporal and spectral properties.

Audio signal can be decomposed in two signal parameters:

- Time and frequency analysis;
- Spectrogram image. analysis

2.2.2.1 Time Domain Analysis

Time Domain Analysis is the field of study of physical signals showing changes through time by using mathematical functions. Changes can be decomposed to identify multiple components.

The following metrics process an audio segment through a time domain analysis:

- Zero-Crossing rate: Measures the number of times the signal waveform changes sign in the course of the current frame.
- Signal power: It is the sum of squares of the signal value normalized by the signal.
- Entropy: Discrete random variable X with possible values x_1...., x_n and probability function P(X).

2.2.2.2 Frequency Domain Features

Time series are composed of a combination of oscillations at different frequencies. It can represent the collective of sinusoids, each one respectively having a specific frequency, amplitude and phase shift (delay factor).

The mechanism to measure these values for any timestamp, is called the Discrete Fourier Transform (DFT). Which converts the time series from the time domain to the frequency domain. From there, identifying amplitudes and phases becomes clear, specially compared to time domain involving heterogeneous oscillations.

After all this pre-processing and DFT application, it is possible to know the average and standard deviation over all frames using the following metrics:

- Spectral centroid: Indicates the location of the "centre of gravity"of the spectrum. It is calculated as the weighted mean of the frequencies. The magnitudes of the frequencies are interpreted as weights.

- Spectral spread: Represents the deviation from the Spectral Centroid.
- Spectral flux: Measures how quickly the power spectral changes. Similar movements should have a deviation of 0 flow.
- Spectral roll-off: Pinpoints frequency below which a specified percentage of the total spectral energy lies, e.g. 85%.
- Spectral entropy: Similar to entropy in the time domain. In impact audio low values are registered while the energy disperses on sub-bands.

MFCC is a technique that calculates a unique coefficient to a particular sample. It is useful to analyse abrupt changes in the spectrum and is commonly used as a main feature in Automatic Speech Recognition.

The steps involved to compute MFCC are the following:

- Pre-emphasis: Emphasize higher frequencies. Increases the energy of the signal at higher frequency.
- Frame blocking: Segment the block in smaller frames. Framing is required as audio is a time varying signal, but when it is inspected over a short period of time, the main properties remain stationary.
- Hamming windowing: Multiply each frame obtained with a hamming window in order to keep continuity of the signal.
- FFT: Convert the time domain into a frequency domain. Applying FFT to each frame we get the respective magnitude frequency.
- Triangular band pass filters: Multiply the magnitude frequency response with a set number of triangular band passes to achieve a smooth magnitude spectrum while reducing the size of the features in the process.
- Discrete Cosine Transform (DCT): Apply DCT on the set of log energy E_k retrieved from the triangular band pass filters.

2.2.2.3 Spectrogram image analysis

The spectrogram representation of an audio signal can visually identify characteristics in speech recognition, where the audio is divided into frames and then features are analysed.

In order to evaluate audio, these have a peculiar time frequency representation. With energy concentrated in the smaller frames of spectral components. Making the information retrieved suitable for classification.

Reference	Approach
Sammarco et al. (2018, 2019)	Proposes detection inside a car by applying feature selection from time and frequency audio signal retrieved from the audio spectrogram running on an app. The smartphone's accelerometer is triggered by movement recorded before and after the event.
Al-Máadeed et al. (2018), Mnasri et al. (2020), Li et al. (2018)	Presents an approach for roads and not individual objects. An anomalous detection system is implemented to detect a specific type of audio, mainly tire skidding and car crashes. A Deep Audio Representation (DAR) is extracted by a Deep Autoencoder Network (DAN) and these features are then combined with the classifier of Bidirectional Long Short Term Memory (BLSTM), improving the detection of anomalous audio.
Baumgärtel et al. (2014)	Proposes structure-borne sensors, specifically piezoelectric, attached to the body panels of the vehicle. Those sensors can determine the type, severity and location of the damage. Signals retrieved are sent to a central electronic module, where they are decomposed by using an analysis filter and then compared with benchmark audio.
Stefanakis et al. (2015)	Presents an algorithm for real time detection and classification of impact audio. Relying only on spatial features that exploit the difference in location of each impacted structure with the use of a compact sensor array. Recovered source amplitudes are used for estimating the source activity in time.

Table 2.3: Audio Signal Processing Relevant Literature

2.3 Driving Behaviour

A majority of vehicle accidents are caused by human factors such as irresponsible driving (i.e., aggressive, furious, raging actions). But distractions can also be due to intentional attitudes, like using a smartphone which in turn leads to a concentration lapses while driving but also can be caused by non controllable factors like sudden health conditions that suspends normal consciousness or concentration while driving.

All these conditions can lead to the same end result, anomalous driving. Once an instance of inconsistent driving is detected a mechanism must be applied in order to intervene.

In an automotive context safety is the number one priority for OEMs. When developing a vehicle, new researches and developments contribute to improving safety. To detect anomalous events, multiple parameters need to be collected, mainly data from sensors, that can register such events and an algorithm must analyse and process the retrieved data so it can deliver a reliable classification.

Sensing can be deployed using various sensors present in a vehicle, i.e., Anti-Lock Brake System (ABS), ESP, that already detect anomalous events and try to suppress the side effects while maintaining control of the vehicle.

Sensors carried in our premises can be easily found in smartphones. Nowadays a low entry smartphone offers accelerometers, gyroscopes sensors and provides speed and location via GPS signal. Data obtained from this sensors give a clear overview on the whereabouts regarding vehicle behaviour.

After all these parameters are stored, an algorithm processes the input data and classifies whether the event is anomalous or not.

Reference	Approach
Cheng et al. (2018)	Delivers an ANN proposition to classify behaviour in efficiency driving.
Cai et al. (2018), Campo et al. (2018)	Uses Extreme Learning Machine (ELM) to classify driving behaviour and identify the driver according to their driving profile.
Saleh et al. (2017), Mumcuoğlu et al. (2019), Mantzekis et al. (2019), Moukafih et al. (2019) and Savelonas et al. (2020)	Use RNN with special focus on LSTMs to classify the driving behaviour.
Matousek et al. (2018) use SVM and Vaitkus et al. (2014)	Presents an algorithm which processes features via Sequential Forward Feature Selection (SFFS) to classify driving behaviour.

Table 2.4: Driving Behaviour Relevant Literature

2.4 Impact Detection

To complement the in-depth review, for each topic a statistic analysis provides a cleaner view on techniques and algorithms used.

Each topic analysis will consist of graphs illustrating a simple overview to later discuss the results achieved during the search phase.

A description is associated with every graph, giving a synopsis on what can be observed and retained. The majority of the graphs presented are column or pie data charts to differentiate all component's respective presence in terms of frequency.

2.4.1 Inertial Sensors

This topic contains a diversity of information thanks to an extended research. Where the only information lacking are the algorithms performance results. Very few references provided results related to the algorithms, this happens due to the fact that most of them only intend to detect an event and not classify it against real cases.

2.4.1.1 Inertial Sensors

This subsection illustrates the hardware used in the references analysed.

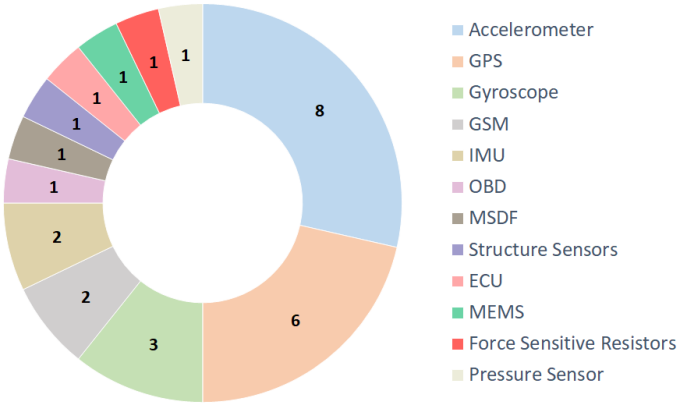


Figure 2.1: Inertial Sensor Studies Frequency of Sensors Pie Chart

Instantly, accelerometer stands out from the rest, followed up by GPS and gyroscopes. With other sensors being less crucial but still having a presence in references for being used to detect impacts.

From a total of 28 sensors selected, the top 3 sensors make up 60% of the total of occurrences.

2.4.1.2 Inertial Sensors Features

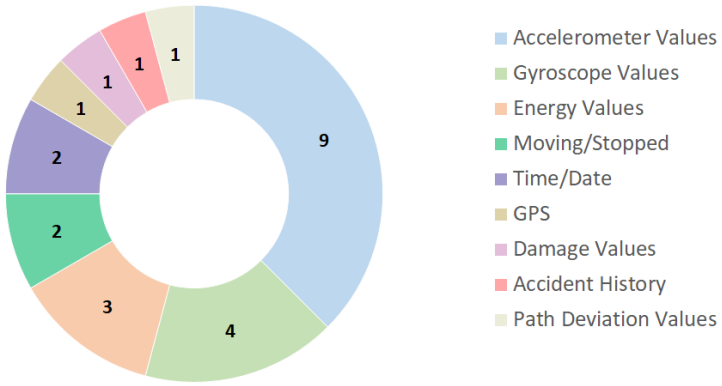


Figure 2.2: Inertial Sensors Studies Frequency of Features Pie Chart

Regarding features, accelerometer values stands out from every other feature used. With nine occurrences in total showing it as a predominant source of data for processing.

A peculiar feature appearing is related to the works of Ali et al. (2015), who uses accident detection history record as a feature when detecting an impact. Drivers with accidents registered in the past may trigger a positive impact classification when compared to drivers that have a cleaner history record.

2.4.1.3 Inertial Sensors Algorithms

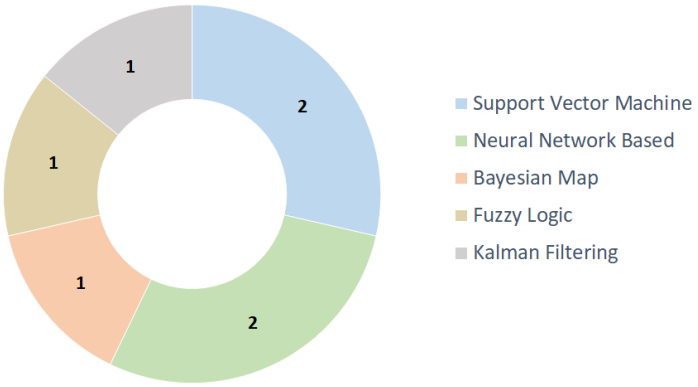


Figure 2.3: Inertial Sensors Studies Frequency of Algorithms Pie Chart

The distribution of the algorithms used in the references analysed turned out to be balanced.

2.4.2 Audio Signal Processing

In this topic, the number of articles analysed turned out to be shorter than expected. The research in this area related to the keyword "Impact Detection"is limited, some of the references studied, i.e., Foggia

et al. (2016), Li et al. (2018) and Mnasri et al. (2020) propose hypothesis for surveillance of roads and not individual objects. On the other hand McLoughlin et al. (2015) focuses on audio events not associated with road users or vehicles.

Related to vehicles, Sammarco et al. (2018) goes further than the rest by focusing on direct impact detection.

Baumgärtel et al. (2014) also localizes impacts using *piezoelectric* sensors.

Despite the number of references appointed being lower than expected, most of them have appropriate material for research purposes. The only information lacking is sensors used for data collection, as research in this area uses publicly available datasets. Some of them released by universities or in the case of Mnasri et al. (2020) that uses YouTube videos as a reference point to work on.

2.4.2.1 Audio Signal Processing Sensors

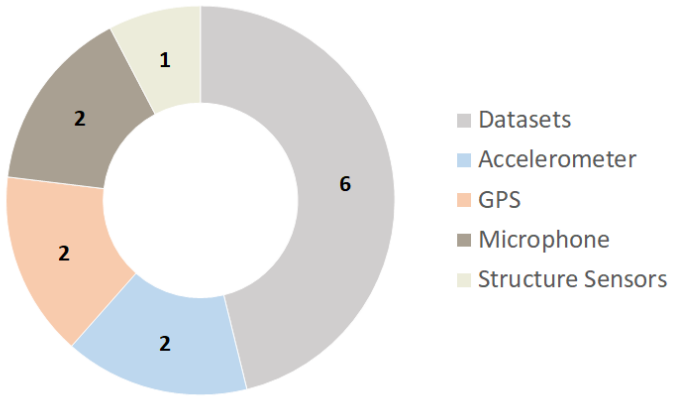


Figure 2.4: Audio Signal Processing Studies Frequency of Sensors Pie Chart

Two approaches are made to collect data, either using publicly available datasets or collecting own data via sensors. After observing the data, the split between using already available datasets or collecting new data have a identical percentage with collecting data having a edge over using available datasets.

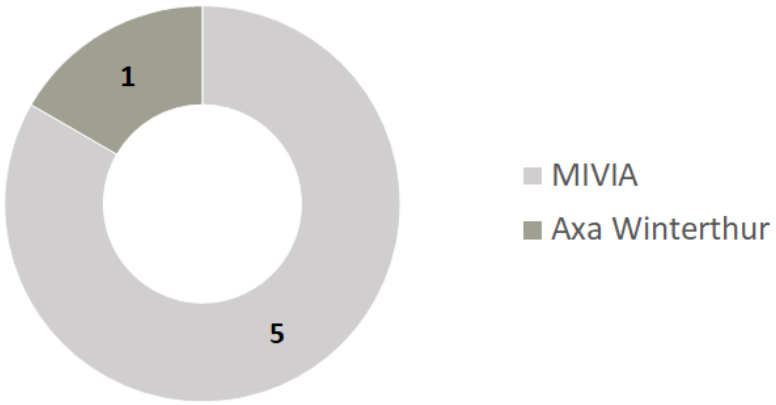


Figure 2.5: Audio Signal Processing Studies Frequency of Dataset Pie Chart

From the pie chart displayed above is noticeable the difference between the data used. The fact that MIVIA is publicly available while Axa Winterthur is not helps explain the difference.

MIVIA is a research lab in the University of Salerno. This dataset contains 400 events, 200 car crashes and 200 tire skidding providing an extended source of audio to process and classify events.

Axa Winterthur, which is used by Sammarco et al. (2019) for being directly involved with the insurance company AXA. A total of 46 audio signals belonging to a crash impact make up part of the dataset.

2.4.2.2 Audio Signal Processing Features

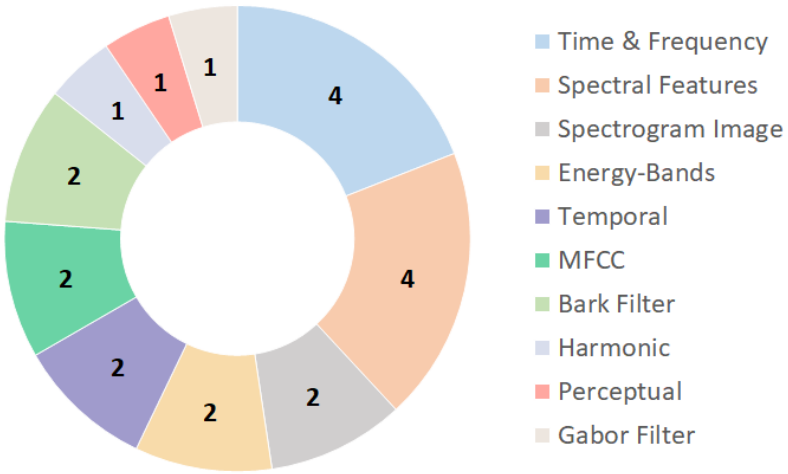


Figure 2.6: Audio Signal Processing Studies Frequency of Features Pie Chart

Ten different features are displayed in the previous pie chart. Time & frequency and spectral features appear with the highest number of occurrences, due to the fact these segments are crucial components of audio. The remaining features also have some relevance in audio composition but not in the same amount.

2.4.2.3 Audio Signal Processing Algorithms

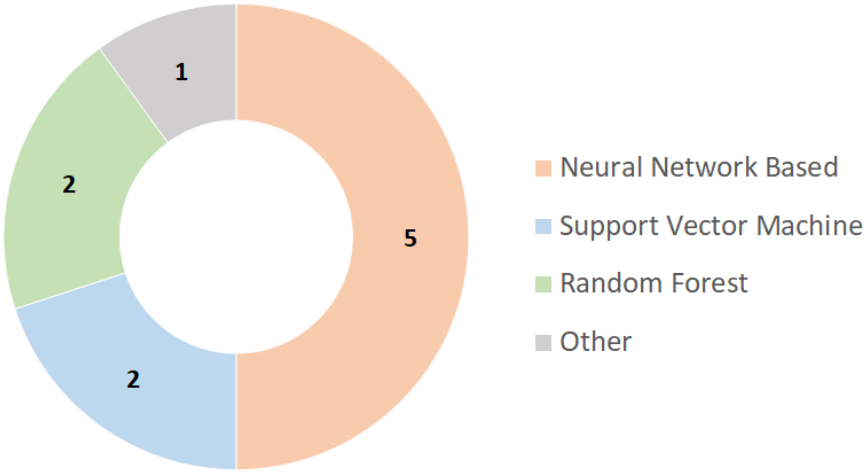


Figure 2.7: Audio Signal Processing Studies Frequency of Algorithms Pie Chart

Half of the total algorithms reported belong to NN algorithms. SVM and RF share between them the same number of occurrences.

The following pie chart displays the occurrence of each NN algorithm.

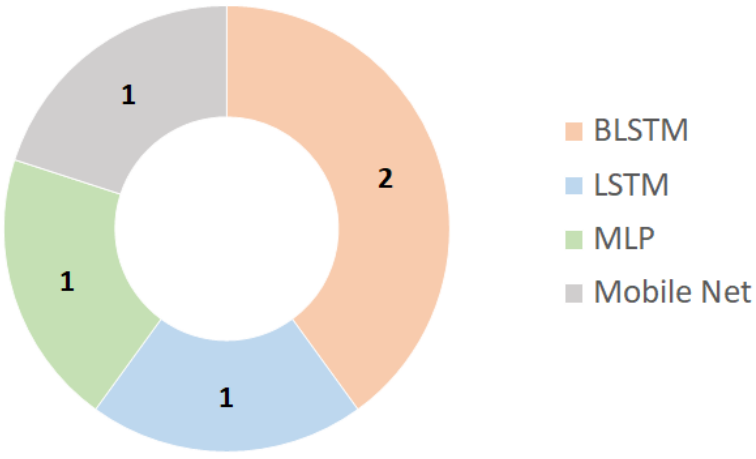


Figure 2.8: Audio Signal Processing Studies Frequency of Neural Network Algorithms Pie Chart

2.4.3 Overview

For the topic Impact Detection, the previously detailed sub-topics Audio Signal Processing and Inertial Sensors offer a modernized path to detect an impact and in some cases classify such events. After doing

an exploration and analysis for the respective sub-topics, there is the possibility of displaying similarities between sub-topics.

2.4.3.1 Impact Detection Sensors

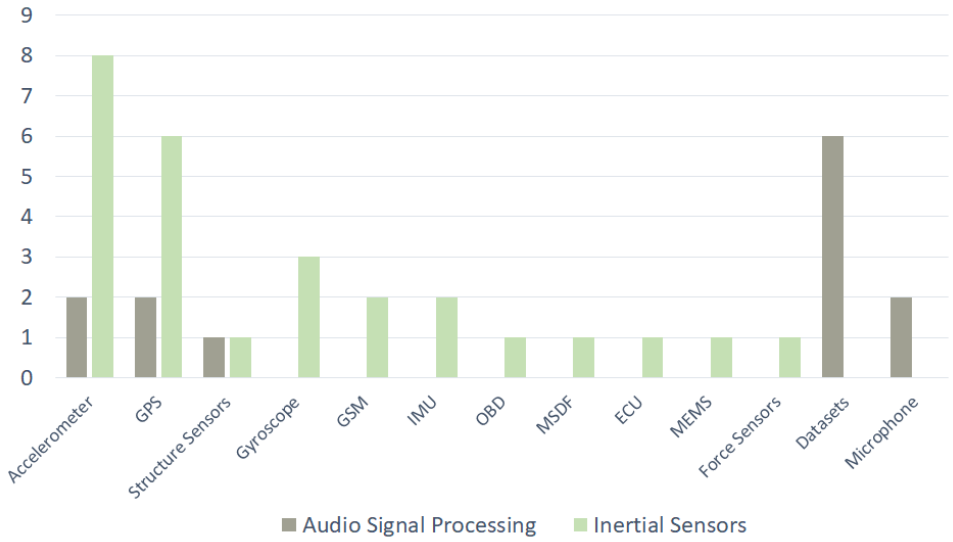


Figure 2.9: Impact Detection Studies Frequency of Sensors Column Chart

Twelve different sensors are taken into account but only three of them are common to both sub-topics. For two of them, accelerometers and GPS, the difference in occurrences is noticeable.

2.4.3.2 Impact Detection Features

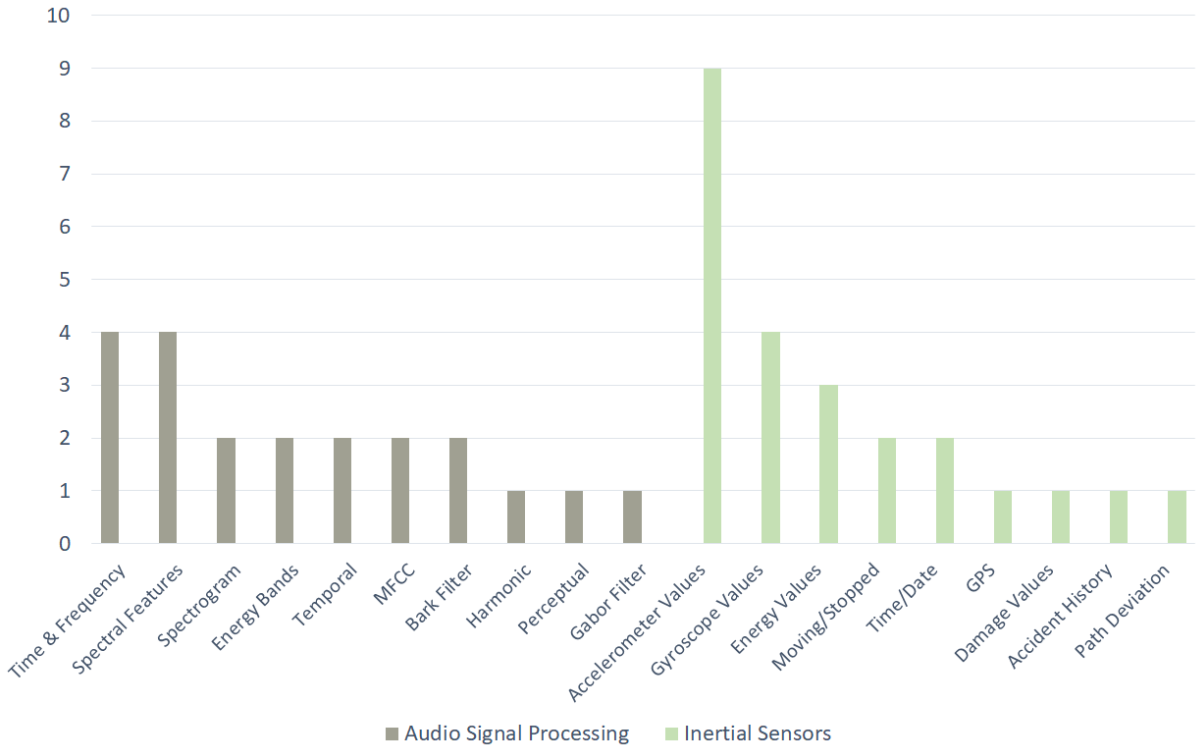


Figure 2.10: Impact Detection Studies Frequency of Features Column Chart

A total of nineteen individual features with zero correlation between sub-topics.

The information provided by the column chart represented in the figure 2.10 exposes how different both sub-topics are when processing data for the same purpose.

2.4.3.3 Impact Detection Algorithms

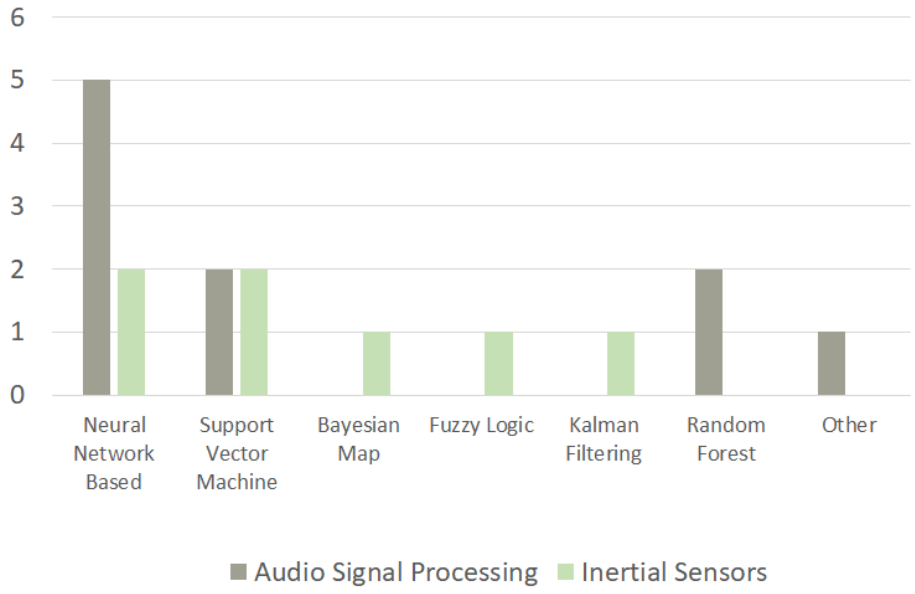


Figure 2.11: Impact Detection Studies Frequency of Algorithms Column Chart

NN algorithms stand out for being employed by both sub-topics with a total of seven instances, In Audio Signal Processing, the use of NN is remarkable when compared to Inertial Sensors.

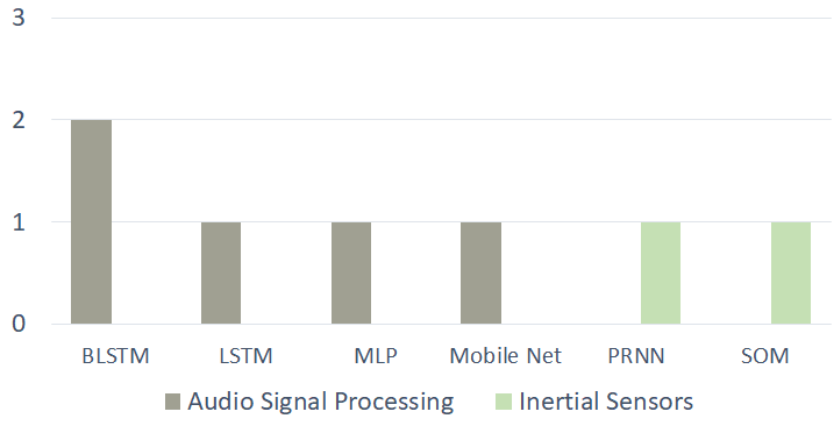


Figure 2.12: Impact Detection Studies Frequency of Neural Network Column Chart

When inspecting the figure 2.12, the seven NN algorithms noted do not share sub-topics.

2.5 Driving Behaviour

Thanks to an extended research, a large amount of information was retrieved related to this topic. The previous topics only enter in action when an impact has occurred. Driving behaviour takes into account

every moment preceding an impact, any style, action, thought is stored for later interpretation in case an impact has occurred and whether the driver is responsible or not.

The following graphs display results observed during research.

2.5.0.1 Driving Behaviour Sensors

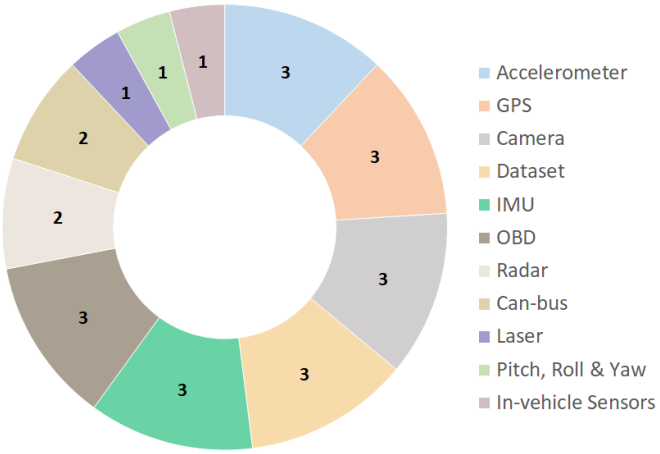


Figure 2.13: Driving Behaviour Studies Frequency of Sensors Pie Chart

As previously observed, accelerometer sensors have a strong influence in data collection hardware.

Radar also has a pertinent presence denominating the existence of new technologies. Not relying only on changes in movement but also providing a video interpretation of the actions.

2.5.0.2 Driving Behaviour Features

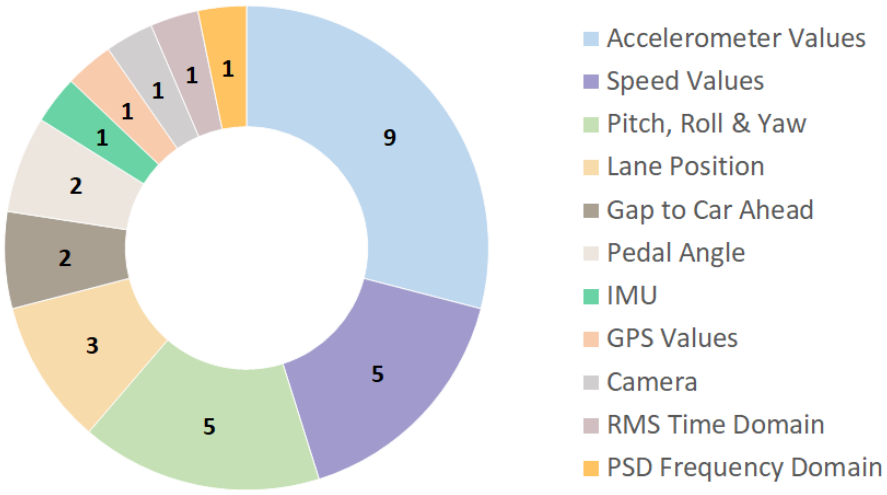


Figure 2.14: Driving Behaviour Studies Frequency of Features Pie Chart

Accelerometer values are the most used feature overall followed closely by speed values and pitch, roll & yaw.

2.5.0.3 Driving Behaviour Algorithms

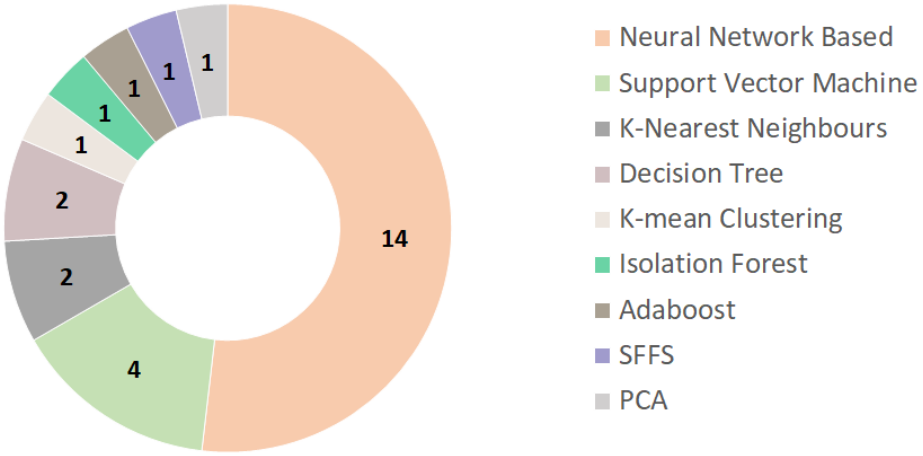


Figure 2.15: Driving Behaviour Studies Frequency of Algorithms Pie Chart

After analysing the algorithms used in the references, approximately $\pm 52\%$ are NN. Once again, when dismantling NN based algorithms, LSTM display the highest number of occurrences.

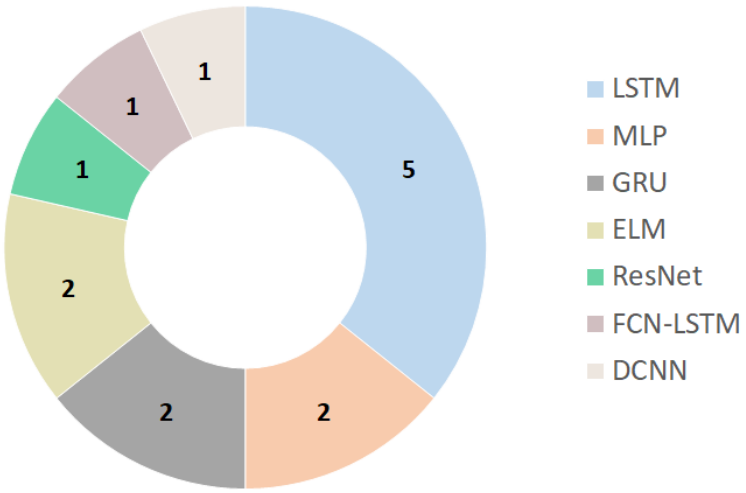


Figure 2.16: Driving Behaviour Studies Frequency of Neural Network Algorithms Pie Chart

2.6 Results

After an analysis was made on the topics, Impact Detection and Driving Behaviour, both are subjected to an evaluation on the performance achieved. Most of the references analysed were presented with a

results/evaluation section.

The top metrics analysed are accuracy and f1-score. When reviewing references, the only one who provides both metrics for the same algorithm is Li et al. (2018). Any other reference that presents results is only under one metric.

The mathematical representation of accuracy is the following:

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

Where True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN).

For F1-score the mathematical representation is the following:

$$F1 - score = 2 * \frac{Precision * Recall}{Precision + Recall}$$

Where $Precision = \frac{TP}{TP+FP}$ and $Recall = \frac{TP}{TP+FN}$. Both metrics have distinct values due to the different parameters used to calculate them. Accuracy is used to measure correctly identified cases, but works best when all classes are equally important. But lacks in performance when classifying FP and FN.

F1-score steps in to minimize the error in this cases, which can affect performance at a serious level. F1-score works better on unbalanced classes. With distribution not favoring one class at all. Working overall as a better metric to evaluate the performance but at serious decrease in score when compared to accuracy.

2.6.1 Impact Detection

2.6.1.1 Results Inertial Sensor

The lack of results presented in this sub-topic is directly shown in the table that follows. Only three references provided results, in the form of accuracy.

Reference	Accuracy	F1-Score
Pai et al. (2014)	✖	✖
Yee et al. (2018)	✖	✖
Nath et al. (2018)	✖	✖
Supriya et al. (2020)	✖	✖
Gontscharov et al. (2014)	✖	✖
Naranjo et al. (2009)	✖	✖
Parviainen et al. (2014)	✖	✖
Selmanaj et al. (2017)	✖	✖
Cismas et al. (2017)	✖	✖
Ali et al. (2015)	✔	✖
Surakul et al. (2016)	✔	✖
Faiz et al. (2015)	✔	✖

Table 2.5: Inertial Sensor Available Results

The table 2.6 details the results in the form of numeric values.

Reference	Accuracy	Case
Surakul et al. (2016)	95%	Overturning Detection
	100%	Path Deviation
Ali et al. (2015)	98.67%	Impact Detection
Faiz et al. (2015)	99%	Impact Detection

Table 2.6: Inertial Sensor Accuracy Results

2.6.1.2 Results Audio Signal Processing

In the table 2.7 it's observable that all but one reference presents results. Two references present both metrics as an evaluation.

Reference	Accuracy	F1-Score
Sammarco et al. (2018)	✔	✖
Sammarco et al. (2019)	✔	✖
Baumgärtel et al. (2014)	✖	✖
Al-Máadeed et al. (2018)	✔	✖
Mnasri et al. (2020)	✔	✔
Li et al. (2018)	✔	✔
Foggia et al. (2016)	✔	✖
Foggia et al. (2019)	✔	✖

Table 2.7: Audio Signal Processing Available Results

To have a better overview on the results component there is a need to differentiate algorithms and metrics, only that way, are they comparable. For each metric, a table is created to display every score associated and the algorithm used. Starting with the accuracy, the following are the results achieved by the authors.

Reference	Accuracy	Algorithm
Sammarco et al. (2018)	87.50%	Random Forest
	90%	Random Forest
Sammarco et al. (2019)	84%	Random Forest
	86%	Random Forest
Al-Máadeed et al. (2018)	93%	SVM
Mnasri et al. (2020)	96%	MLP
	97%	LSTM
	97%	BLSTM
Li et al. (2018)	92.50%	BLSTM
Foggia et al. (2016)	78.95%	SVM
Foggia et al. (2019)	99.50%	Mobile Net

Table 2.8: Audio Signal Processing Accuracy Results in Detail

A total of seven references present accuracy as a metric. The best value achieved was by Foggia et al. (2019), with an 99.5% using Mobile Net algorithm. The lowest value was reported by Foggia et al. (2016) using MLP.

Sammarco et al. (2018, 2019), presents in each reference two scores for RF. The reason why the same algorithm has two different scores is due to in how the data is analysed by the algorithm. The first algorithm focuses on time and frequency components in the audio, reaching an 87.5% score in 2018 and an 84% in 2019, showing a downgrade in performance. The second algorithm is only used in 2018 and focuses on a spectrogram image model calibrated on the detection of percussive, high energy and hollow audio corresponding to impacts. This allowed to reach a score of 90%. In 2019, the score of 86% is the combination of the two algorithms not achieving a high score like in 2018 but placing it between the two algorithms.

Mnasri et al. (2020) stands out from the rest by providing various NN algorithms for the same input data displaying a range from 96% up to 97% score.

SVM accuracy results have a large gap between the maximum and minimum, 93% in Al-Máadeed et al. (2018) and 78.95% Foggia et al. (2016). The adoption of NN by Foggia et al. in 2019 allowed to jump to 99.50%.

In addition to the accuracy metric. f1-score is also present, although in a lower number of references; the scores of Mnasri et al. (2020) and Li et al. (2018) are presented in the table 2.9.

Reference	F1-score	Algorithm
Mnasri et al. (2020)	91.80%	MLP
	92.30%	LSTM
	91.70%	BLSTM
Li et al. (2018)	91.86%	BLSTM

Table 2.9: Audio Signal Processing F1-score Results in Detail

Once again Mnasri et al. (2020) presents multiple results for the same algorithm. Compared to accuracy results, this time f1-score presents lower values but consistent across the range.

2.6.2 Driving Behaviour

In this topic the number of available results is average. Despite having both metrics present, entries containing both metrics for the same reference are nonexistent. Table 2.10 presents the available scores provided and the metric used.

Reference	Accuracy	F1-Score
Ly et al. (2013)	✖	✖
Vaitkus et al. (2014)	✔	✖
Matousek et al. (2018)	✖	✖
Saleh et al. (2017)	✖	✔
Mumcuoğlu et al. (2019)	✔	✖
Mantzekis et al. (2019)	✔	✖
Feng et al. (2018)	✖	✖
Moukafih et al. (2019)	✖	✔
Cheng et al. (2018)	✔	✖
Cai et al. (2018)	✔	✖
Campo et al. (2018)	✔	✖
Savelonas et al. (2020)	✔	✖

Table 2.10: Driving Behaviour Available Results

Reference	Accuracy	Algorithm
Vaitkus et al. (2014)	100%	SFFS, KNN
Mumcuoğlu et al. (2019)	92.80%	LSTM
Mantzekis et al. (2019)	78%	LSTM
	84%	GRU
Cheng et al. (2018)	93.60%	Feed Forward
	92.70%	Feed Forward
Cai et al. (2018)	92.50%	Deep Convolutional Neural Network (DCNN)
	85%	SVM
	95%	ELM
Campo et al. (2018)	95%	ELM
Savelonas et al. (2020)	95%	GRU

Table 2.11: Driving Behaviour Accuracy Results in Detail

Having a generous range of accuracy values adds to a better analysis on the algorithms used. The range of values reaches a maximum of 100% in Vaitkus et al. (2014) and a minimum of 78% in Mantzekis et al. (2019).

The works of Mantzekis et al. (2019) and Cai et al. (2018), are relevant for testing and evaluating more than one algorithm. This additional step allows to understand and differentiate algorithms with the same input data.

Reference	F1-score	Algorithm
Saleh et al. (2017)	48%	MLP
	80%	Decision Tree
	91%	LSTM
Moukafih et al. (2019)	94.11%	Random Forest
	92.75%	Adaboost
	88.29%	ResNet
	85.22%	LSTM
	95.88%	Fully Convolutional Long Short Term Memory (FCN-LSTM)

Table 2.12: Driving Behaviour F1-score Results in Detail

Saleh et al. (2017) and Moukafih et al. (2019) are the only authors who presented results under the f1-score metric, and for each one multiple algorithms were tested.

2.6.3 Conclusion

After exploring every single result regarding each component, the next step is to make a conclusion on what is important to take on from here for the development and implementation of an algorithm.

2.6.3.1 Sensors

For inertial sensors the following sensors are fundamental, accelerometer, gyroscope and IMU. For audio signal processing the fundamental sensor is a microphone.

2.6.3.2 Features

The most relevant features for audio are time & frequency, sepctral features and spectrogram.

For Inertial sensors, data retrieved by accelerometers and gyroscopes are the most useful sources of data to rely on.

2.6.3.3 Algorithms

It was easy to observe the sheer presence of NN returning amazing scores. The author Sammarco et al. (2019), mentions that all new approaches to detect impact in the audio domain are starting to use NN algorithms.

The NN algorithms that stand out from the rest are LSTM and BLSTM.

After analysing the topic Driving Behaviour, which purpose is not to detect impacts, there was a strong presence of algorithms in the NN domain. The recent relevant literature is focused in these algorithms due to their state of the art performance.

Chapter 3

Problem statement and proposed solution

3.1 Problem statement

In the scope of this thesis, the main objective is the development of algorithms based on deep learning techniques capable of detecting and classifying impacts that inflict damage on a vehicle. The objective is to achieve the goals defined in the Introduction chapter.

For the first objective the data to be used comes from an accelerometer placed inside a vehicle that constantly registers acceleration values. Through this data, a neural network must be able to detect patterns that identify damage situations. The definition of damage in this scenario are physical anomalies to a vehicle structure, i.e., small dents and broken parts caused by impact.

The research carried out in the State Of The Art chapter, in the impact detection domain allowed to highlight techniques with better performance than other techniques within the deep learning domain. Several NN have been studied and compared for the same type of data. This allowed the identification of advantages and disadvantages for the type of data to be classified.

The data from the accelerometer can be represented by a time series. The research carried out allowed to define that there are NN with better capabilities in discovering patterns compared to other NN algorithms.

For the second objective, which involves going through the AI development cycle, the effort to ensure that this process is carried out involves addressing two challenges so that all phases are carried out.

3.2 Proposed solution

For the first objective the steps start with understanding and analysing existing data. From this point on, the algorithms to be developed are in the domain of deep learning depending on what type of data is to be classified.

The first network to be developed is a MLP trained on features extracted from the accelerometer signal. This involves analysing and extracting the most relevant features for identifying damage and non-damage patterns.

The second network to be implemented is a CNN with 1 dimension based on the signal values. This dimension is more suitable for time series data

The last network to be developed is a RNN. The investigation carried out in the chapter State Of The Art showed that, in comparison with the previous ones, these specific networks behave efficiently with time series data. The most recent papers focused on these networks. Like CNN, these networks are based on signal values. Only the MLP network is based on the extracted features and not on the signal itself.

The second objective is to be able to cycle through all phases in an AI development project. The first challenge is to plan data collection of from audio and gas sensor data.

Once the planning and collection of audio and gas data has been carried out. The remaining phases of the development cycle are within the scope of the thesis. Since the data has already been collected it was impractical to go through all the phases on the main project.

In this way it is possible to fulfil this objective even if they are challenges with different purposes.

3.3 Thesis methodology

Although agile methodology is being implemented in this project the core of the project is ML which involves data science. To work in this area the following methodologies are more suited for a ML project:

- Sample, Explore, Modify, Model, and Assess (SEMMA);
- Knowledge Discovery in Databases (KDD);
- Cross Industry Standard Process for Data Mining (CRISP-DM).

The methodology being used in this thesis is CRISP-DM, for being an extended version of KDD, but also, for being familiar.

3.4 CRISP-DM Phases

The following sub-sections give an in depth tour through all the phases and respective components.

3.4.1 Business Understanding

The first phase is identifying the problem at hands, considering the actual resources available and if is a viable project. In this phase it is required to define objectives, data mining and work plan.

- Goal definition: Translate the problem into a corporate scene. Establishing a client/business relationship. Defining what components are being introduced in the market while analysing and comparing products from rival companies.
- Data mining: Decompose the problem into objectives. Specifying the problem at hand while describing expected outputs and how they should be achieved.
- Work plan: Define a starting process, respective execution while identifying the top objectives and technical aspects to be used to answer the problem.

3.4.2 Data Understanding

The fist encounter with the raw data, after collection and processing via a chosen program.

- Data collection: Illustrate in a translucent way the data source and how the data was collected. If data access was public or private, and if so, what credentials were used to access the data in question.
- Data exploration: Exploring the data in general, checking what can be the main attributes, the type of values being observed and their frequency.
- Work plan: Define a starting process and respective execution while identifying the top objectives and technical aspects to be used to answer the problem.
- Data quality: Answering questions about the data, if there are any missing values, if the values are in an acceptable range and if any anomaly was detected how frequent they were.

3.4.3 Data Preparation

Data to be analysed is selected.
This key factor requires time and resources during the implementation task.

- Data selection: Choosing and analysing data from a rational point of view, while at the same time explaining the logic behind it.
- Data clean: Remove any unwanted data from the subset in order to achieve a better data quality. Modulation technical applications are recommend for this step. This actions must always be justified.
- Data construction: Derive attributes and register creation.
- Data integration: Merge data, from the initial subset with the new one, in a correct and coherent way, this can be achieved by merging or aggregating both parts.
- Aggregation: Aggregate values from multiple registers and/or tables and join them in a solo entry.

3.4.4 Modelling

This phase involves selecting the modelling process to be implemented. A clear idea on what implementation should be used already has been discussed, however this is where a detailed model should be chosen, for example, linear regression, hierarchical clustering, neural networks, among others.

More than one model can be selected, but each one has to be implemented and explained separately from the other ones. Interpreting the model and acknowledging results, should be take into account to create a rank of models based on their performance and/or complexity.

- Modelling procedure: Document what procedure is being used.
- Modelling statements: Explain every approach and adaptation used in data and models.
- Prototype developing: Test and evaluate a model prototype's quality before proceeding to the next phase. Using specific datasets in order to train and test the model and consequently validating if it is worth it or not.
- Parameter definition: Explore and adapt the different parameters in use, for a better data incorporation.
- Models - Create the first model using a tool system in a practical approach and not a theoretical one.
- Model description: Mark the results achieved after testing, and interpret them accordingly with expectations while also pointing out encountered problems.
- Model review: Summarize the results accomplished in this assignment while pointing out the pros and cons throughout the performance delivery.
- Model parameters review: Modifying at each iteration the model parameters previously defined in order to gain better performance.

3.4.5 Evaluation

After collecting all results achieved. A deep analysis should be made based on all results observed and consequently dictate what is the best model, inquiring the disadvantages, and how they can affect the following project phases. Performance aspects are also taken into consideration, poor efficiency and heavy complexity can affect time and money resources in a real time application. If any of these previous steps are dealt with, the process until this point must be reviewed to discard any ignored steps that could affect the project as a whole.

- Data mining result analysis: Compare results obtained with those planned at the beginning of the project and if they meet the requirements.

- Approved models: Approve every model tested that meets criteria and requirements for future phases.
- Process review: Sum up all steps incorporated in the process while reviewing all activities that led to it.
- Possible actions: List all possible actions properly justified to apply in the next steps.
- Decision: Describe how to proceed to next phase in a rational way.

3.4.6 Deployment

The final phase in the CRISP-DM methodology, is to deploy the final product into the client business. This phase should be monitored in closely by the development team, and be supported with documentation and a maintenance plan throughout all implementation. This is a crucial phase in the project affected not only the developments made, but also by the market that will use it also has impact on the project. In the end, form an analysis review about the project as a general while pointing out the positive, negative and average points.

- Implementation Plan: Summarize the strategy to apply and all steps to achieve it.
- Support & Monitoring: Define a support and maintenance strategy.
- Final Report: Write a report containing all aspects related to all steps, from planning to deployment.
- Final Presentation: Present the project to the client, with results to back it up.
- Documentation: Indicate and mention every single item during development of the project in a document.

Chapter 4

Data collection for detection of anomalies in a vehicle's cockpit project

After all the research work done and the presentation of the problem statement. It is time to start the machine learning project cycle, which essentially consists of the following steps in accordance to CRISP-DM cycle presented in the chapter 1.

- Collecting Data;
- Preparing data;
- Choose a model;
- Train the model;
- Evaluate the model;
- Hyperparameter tuning;
- Predict.

In this chapter only the initial steps were performed within the scope of a set of audio and gas sensors. The remaining steps were performed in the next chapter.

4.1 Data collection planning

This work was done in a joint partnership between University of Minho, more specifically, the Centro de Computação Gráfica and Bosch. There was a requirement to acquire data from two components, audio anomalies and air quality inside a vehicle. These requirements where set out in a document shared by both parties.

After knowing what would need to be collected and the quantity it became clear that this would only be possible with hardware and software support prepared for this data collection. For this purpose a setup consisting of a microphone, a particle sensor and a gas sensor were used. Multiple sensors were built with this configuration in order to cover several vehicles at the same time. Increasing the performance of the collection and potential dissimilarity detection of differences between the vehicles.

The use cases that would have to be included in the collection were the following:

- Normal Wake state: Normal sounds inside a vehicle without anomaly events.
- Talking: Normal conversation between occupants.
- Texting and talking: Passenger uses mobile phone during the trip, may also include calls
- Singing: Passengers sing to the rhythm of a song, driver may be included.
- Cough: Passenger coughs or sneezes.
- Argument: Occupants of the vehicle have an argument.

For each use case, the equivalent of between 3 to 5 hours of data was required according to a list of variants. The variants are a combination of open/closed windows, radio on/off. Of the total time for a use case, 70% of the data collected would have to be with windows closed and radio off while the remaining 30% would have to be distributed over the remaining possible combinations.

For the sensor setup there was a list of requirements to follow. For the microphone, the sampling rate would be 44100Hz on a single channel and if recordings were to be segmented, each segment would have to be at least 60 seconds long.

4.2 Data collection execution

Once we have defined what to collect and how to do it, the next phase which is execution.Taking into account all combinations and variants according to the use case, we then proceeded to a data collection cycle.

The first cycle of data collection was on stationary vehicles. Throughout this first iteration, errors were detected and subsequently corrected. Microphone configuration errors were the most frequent during this iteration, more specifically in terms of dynamic detection and not fixed detection.

For each event the various attributes relevant during collection were recorded in a document containing the status of each variant during that specific event. The attributes are the following:

- Experiment ID: Identifier of the experiment in each given day.
- Event ID: Identifier of the event (e.g. 0101).
- Event description: Name of the event (e.g. Normal Wake State) .

- Run: Identifier of the exercise.
- Car: Brand and model of the vehicle used for the data acquisition exercise.
- Fuel : Vehicle type fuel.
- Type of road: Type of road of the data acquisition exercise (e.g. asphalt).
- Windows: State of the windows (open or closed).
- Windows description: Description of the state of windows (e.g. front windows open).
- Radio: State of the radio (on or off).
- Radio description: Description of the state of the radio (e.g. radio intensity: low).
- AC: State of the air conditioning (on or off).
- AC intensity: Description of the state of the AC (e.g. AC intensity: low).
- Wipers: State of the wipers (on or off).
- Wipers intensity: Description of the state of the wipers (e.g. wipers intensity: low).
- Passengers: Number of passengers on the vehicle.
- Passenger action description: Description of the passengers events and/or location (e.g. driver and front passenger).
- Average speed: Average speed during the exercise.
- Location: General location where the exercise was performed.
- Obs: Observations and/or new information that is not described on the other specifications.
- Done: If the exercise was completed (1) or it was just planned (0).

4.2.0.1 Sensor setup assembly

The setup was located on the windshield as represented on the figure 4.1. With the microphone being placed on the middle of it and the particle and gas sensor placed on the left side, leaving a gap between both so that the PM2.5 sensor fan cannot be captured by the microphone.

Figure 4.1: Setup placement inside the vehicle

4.3 Data collection results

Finally after a data collection composed of two phases, stationary and moving. The results obtained were the following.

Event description	Total events	Total events in hours
Normal Wake State	99	2 h 12 min
Talking inside the Vehicle	87	1 h 56 min
Talking or Texting	38	50 min 40 sec
Singing	4	5 min 20 sec
Coughing	64	1 h 25 min 20 sec
Argument	0	0 h

Table 4.1: Stationary event data collection

Event description	Total events	Total events in hours
Normal Wake State	154	3 h 25 min 20 sec
Talking inside the Vehicle	197	4 h 22 min 40 sec
Talking or Texting	135	3 h 0 min 0 sec
Singing	142	3 h 9 min 20 sec
Coughing	152	3 h 22 min 40 sec
Argument	138	3 h 4 min

Table 4.2: Moving event data collection

4.4 Exploratory Data Analysis

After executing the data collection, an analysis on it provides an in-depth overview of the features and conclusions that can be retrieved. This exploratory data analysis concerns the three sources of data: microphone, particle and gas sensor.

For the microphone, the analysis is different when compared to the particle and gas sensor. The source wav file cannot be explored and visualized in the same way as the .csv files related to the other sensors.

4.4.0.1 Audio

Audio was recorded in segments of 80 seconds in time duration.

Using the tool Audacity to explore the audio it is possible to visualize the wave form and spectrogram regarding events within the sound sample.

Beyond just visualizing the audio with the available media, processing the audio can also lead to additional information related to other features, like decibel and intensity scale.

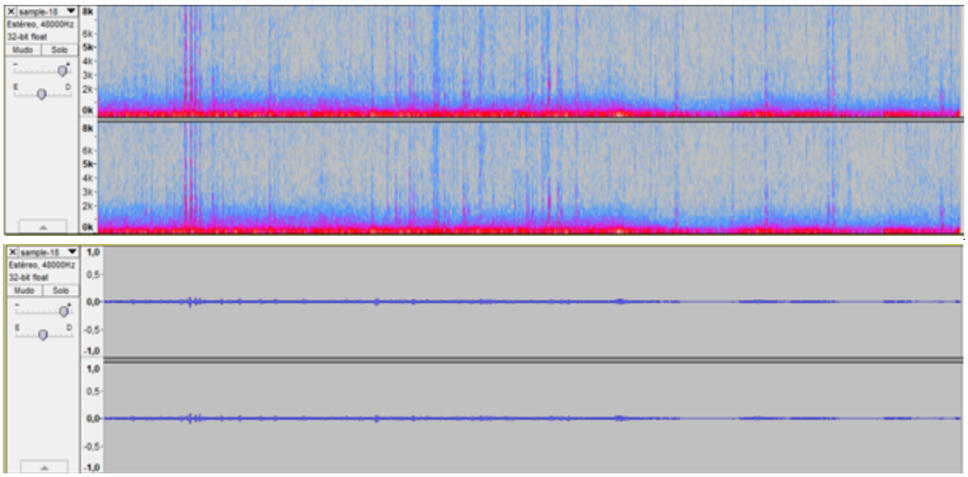


Figure 4.2: Audacity spectrogram and waveform representation

Besides the analysis described above, using scripts mostly made in python allowed for other processing in the same audio samples, providing further information related to events happening within the sound. MFCC represents the power spectrum allowing the association of an event to a decibel and frequency scale.

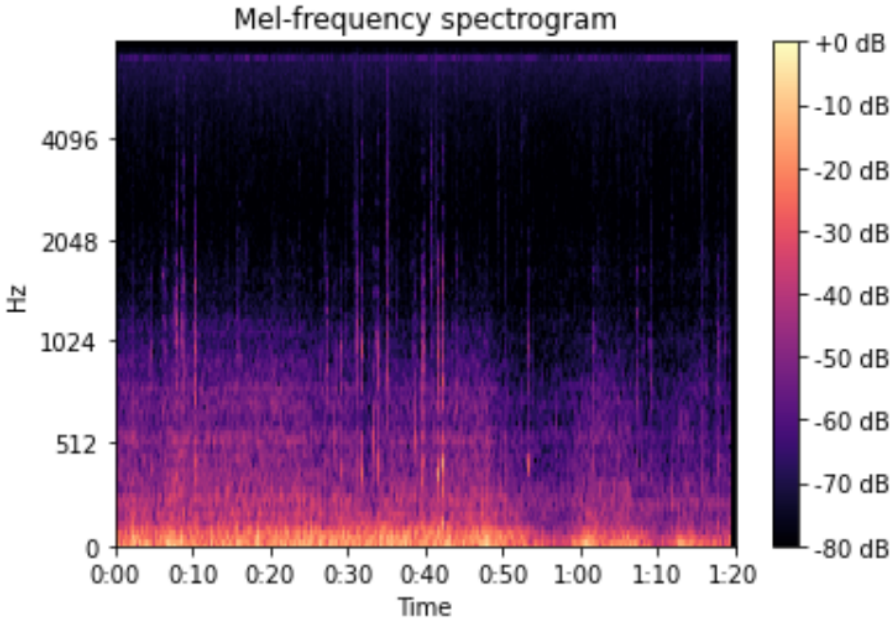


Figure 4.3: Mel-frequency spectrogram representation

4.4.0.2 Particle sensor

The particle sensor (PM2.5) provides multiple attributes related to air quality sensing. The entries observed in the .csv files are the following: pm10 standard, pm25 standard, pm100 standard, pm10 env, pm25 env, pm100 env, 03um particles, 05um particles, 10um particles, 25um particles, 50um particles and 100um particles.

To explore these values, using a script written in python along with the pandas and Matplotlib libraries helps in visualizing values in an organized format.

Pandas allows for a summarized statistical analysis of a .csv file, returning the central tendency. The dispersion and shape of a dataset distribution can be obtained by using the describe() method which for a given run outputs:

	pm10 standard	pm25 standard	pm100 standard	pm10 env	pm25 env	pm100 env	particles 03um	particles 05um	particles 10um	particles 25um	particles 50um	particles 100um
count	480.000000	480.000000	480.000000	480.0	480.0	480.0	480.000000	480.000000	480.000000	480.000000	480.000000	480.000000
mean	8.362500	11.745833	15.727083	0.0	0.0	0.0	1837.895833	719.416667	260.333333	28.060417	5.812500	0.391667
std	1.979501	2.523787	2.719775	0.0	0.0	0.0	389.794756	123.601679	35.186609	3.700490	1.133219	0.571353
min	6.000000	9.000000	12.000000	0.0	0.0	0.0	1449.000000	592.000000	203.000000	22.000000	4.000000	0.000000
25%	7.000000	10.000000	14.000000	0.0	0.0	0.0	1595.500000	632.750000	230.000000	25.000000	5.000000	0.000000
50%	8.000000	12.000000	15.000000	0.0	0.0	0.0	1723.000000	684.500000	259.000000	26.000000	5.000000	0.000000
75%	9.000000	12.000000	17.000000	0.0	0.0	0.0	1968.500000	776.250000	278.000000	31.000000	7.000000	1.000000
max	21.000000	28.000000	31.000000	0.0	0.0	0.0	4930.000000	1565.000000	351.000000	35.000000	8.000000	2.000000

Figure 4.4: Particle sensor describe() method

For a plot visualization, using the Matplotlib library allows for displaying each entry value variation for a given run.

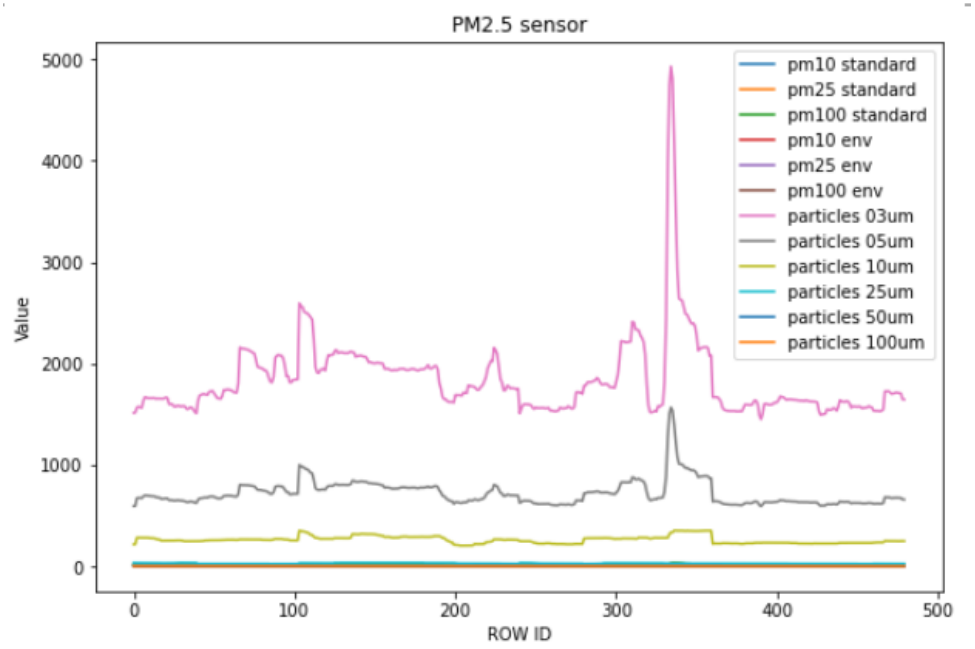


Figure 4.5: Particle sensor plot representation

The X axis is named ID ROW in association to the .csv file row number, and the Y axis refers to the cell value given by each sensor associated with the ROW ID.

4.4.0.3 Gas sensor

This sensor measures relative humidity, barometric pressure, ambient temperature, altitude and gas. The .csv file contains an entry for each measurement.

As explained in the particle sensor Exploratory Data Analysis (EDA), for the gas data, using a script in python along with the Pandas library and using the describe() method outputs:

	temperature	gas_resistence_in_ohms	humidity	pressure	altitude
count	1480.000000	1480.000000	1480.000000	1480.000000	1480.000000
mean	25.195405	273549.470480	27.395932	992.819155	172.541669
std	1.466432	27067.481484	2.206169	17.292307	146.790145
min	23.030000	127717.010000	22.480000	957.540000	-14.180000
25%	24.080000	261962.020000	25.867500	975.665000	25.595000
50%	24.690000	267420.000000	27.530000	997.585000	131.245000
75%	26.402500	286902.000000	28.700000	1010.177500	317.715000
max	27.740000	956580.940000	33.080000	1014.950000	474.500000

Figure 4.6: Gas sensor describe() method

4.5 Conclusion

Despite not being the main focus for this thesis, this side project was important for two reasons; The first one is complementing the initial phase of the CRISP-DM cycle, allowing the process of data planning and collection to be executed.

The second reason was it working as an introduction to how to explore and analyse data. The learning curve deficit encountered in this project helped in adjusting the procedure and approach when facing the main project focus to work on.

The data present in this project, despite being different from the data of the main component, audio and gas sensors versus accelerometer values, allowed for the evaluation of the different sensors and the interaction with other types of data besides the accelerometer values mainly used in the later stages of this thesis.

In conclusion, this project helped not only to understand and complete a ML project cycle but also to explore the multitude of approaches to sensing and the values retrieved by the sensors.

Understanding what was being collected has helped immensely on the completion of this thesis.

Chapter 5

Accelerometer sensor data understanding and analysis approach for damage detection

The primary goal of this thesis is to detect exterior impacts into a vehicle via a sensor setup placed inside. These impacts negatively affect the exterior appearance of the vehicle.

Damage inflicting events can be caused by a person, structure or object (human or non-human made) and can occur while the vehicle is moving or stationary.

The set of sensors used to retrieve information from the vehicle and its surroundings are an accelerometer, gyroscope and microphone. The choice of sensors was based on the fact that most events previously mentioned can be perceived or detected by the forces involved as a result of the impact and by the resulting sound. The IMU records information that allows to perceive the forces applied and microphones allow to capture sounds that result from these events.

This system allows to alert the owner of the vehicle if any anomalous event has been detected on the vehicle while moving or stationary. The elements found in this use case are the following:

- Actor: Vehicle in combination with a sensor set.
- System: A sensor reading multiple parameters capable of detecting damaging events occurring.
- Goal: Detect small damage.

The requirements presented for this system are listed below:

- Detect event resulting in damages to the vehicle and classify the damages in the range of cosmetic, significant and severe damages.
- Determine location of the event generating the damage.
- Determine the type of event generating the damage.

- Take into account the different structures of the vehicles.
- Data retrieved from accelerometer, gyroscope and microphone.

5.1 Data collection

5.1.1 Data collection sensor setup

In order to create a system capable of detecting a damage event there is a need to develop and build a setup capable of executing this task.

Starting with the setup which is composed of multiple sensors and processors, all mounted in a printed circuit board (PCB). For the purpose of this use case there will only be a specific analysis on the accelerometer, gyroscope and microphone sensors. Multiple setups were implemented with the accelerometer and gyroscope, ranging from having them mounted separately or using the IMU to combine both in just one physical sensor unit. Regarding the microphone, two units are mounted in opposite corners of the PCB. The accelerometer and gyroscope specification of each sensor is described below.

	Accelerometer			Gyroscope		
Sensor	Output Data Rate	Bandwidth	Range	Output Data Rate	Bandwidth	Range
BMI270	1600Hz	434Hz	± 8G	1600Hz	134Hz	± 250 deg/s
MPU9250	1000Hz	218.1Hz	± 8G	1000Hz	184Hz	± 250 deg/s
SMA130	1000Hz	500Hz	± 8G	-	-	-
BMG250	-	-	-	1600Hz	523.9Hz	± 250deg/s

Table 5.1: Accelerometer and gyroscope specifications

The IMU BMI270 and MPU9250 were used as accelerometer and gyroscope devices since they comprise both. The SMA130 and BMG250 were used together since they are an accelerometer and a gyroscope, respectively. Only one microphone model was used

Sensor	Sample Rate	Bandwidth
PNs SPG08P4HM4H-1	44100Hz	10000Hz

Table 5.2: Microphone specifications

It was important that the accelerometer and gyroscope data be sampled in synchronization so that information obtained from these could be used in a complementary way if needed without requiring extensive post processing and manual alignment. Since there is a considerable difference between the audio sample rate and the others two sensors, it was acceptable that the synchronization could be within a window of 5ms.

The sensors must be placed on the windshield right next to the rear-view mirror, this choice of position has its advantages but requires that some post processing according to the vehicle be introduced. After

having a setup ready to collect data, a plan was created to outline events that needed to be collected for later analysis.

5.1.2 Data collection event planning

Two main groups were identified: Damaging and non-damaging events. The plan was conducted so that the experiments could replicate as closely as possible real-life situations. Damaging events involved a careful setup since human safety was a priority. In non-damaging events human safety was also a priority, however since there is no impact/damage involved, the risk of injuries is lower when compared to damaging events collection.

For each event, a set of parameters indicates what to consider in doing while collecting data:

- Experiment description: Description of what should be done and how many times it should be repeated, among other recommendations.
- Labelling: Designation of the label to use on the event collected.
- Post process: Adjustments to be made after collection.
- Constraints: Present obstacles that could limit the event collection. This parameter is optional.

The following tables represent the events planned for collection regarding the damage and moving status.

Event ID	Non damaging events
EV01	Door slamming
EV02	Low curb bump with wheels
EV03	Speed bumps
EV04	Stomping occupant/party people
EV05	Trunk lid open/close
EV06	Hood open/close
EV07	Hood slamming
EV08	Sunroof open/close
EV09	Switch pushed in mirror panel
EV10	Roofline slapping
EV11	Sunvisor open/close
EV12	Sunvisor detach/attach
EV13	Make up mirror open/close
EV14	Wiper flapping
EV15	Interior rear view mirror adjusting
EV16	Side mirror folding
EV17	Side mirror collision
EV18	Object placed on the roof
EV19	Wiper activation/deactivation
EV20	Side window opening/closing

Table 5.3: Non damage event data collection planning

Table 5.3 continued from previous page

Event ID	Non damaging events
EV21	Door slamming engine off
EV22	Speed bumps plus ABS braking
EV23	Windshield slapping
EV24	Smartphone/GPS mount fixation
EV25	Object sliding against windshield
EV26	Switch dipping mirror
EV27	Ventilation maximum
EV28	ABS braking event
EV29	ESP intervention
EV30	Rough road
EV31	Belgisch block
EV32	Aquaplaning track
EV33	Engine load change during acceleration
EV34	Engine load change during deacceleration
EV35	Carwash
EV36	High-pressure washer
EV37	Wiper on frozen windshield
EV38	Engine on (auto start/stop)
EV39	Person knocking on the vehicle structure
EV40	Throw object at the car (e.g., a football ball)
EV41	Person touching / shaking the vehicle

Event ID	Stationary damaging events
EV42	Scratching with object across vehicle
EV43	Bump collision (get bumped by another vehicle)
EV44	Side collision with another vehicle (left/right)
EV45	Hitting the car with object (baseball bat/hammer)
EV46	Throw object at the car

Table 5.4: Stationary damage data collection planning

Event ID	Moving damaging events
EV47	Speeding over speed bump/pothole
EV48	Vehicle hits object
EV49	Vehicle side drags on object
EV50	Vehicle drives over obstacle
EV51	Scrape with object when cornering
EV52	Vehicle bumps with rim on low curb

Table 5.5: Moving damage data collection planning

The data retrieved was stored in two separate files: JavaScript Object Notation (JSON) and Hierarchical Data Format (HDF) files, more specifically a HDF5 file. The JSON file contains the metadata referring to the attributes of the event which are the following:

- Car identification
- Driver identification
- Sensor location inside the vehicle
- Event label
- Event start time
- Event end time
- Damage status
- Damage type
- Damage severity
- Road type
- Weather type

Accelerometer, gyroscope and audio data are stored in the HDF file.

5.1.3 Data collection labelling

For each data collection event, the approach on carrying out the labelling has two stages in order to assure that each event is correctly labelled and in the correct time series. The first stage happens in the data collection process, while collecting the data a software tool developed by the work team allows for recording signal data and the labelling in real time. The software tool allows for an association between a planned event and a hot key for a faster label and a lower probability of making mistakes while in the process.

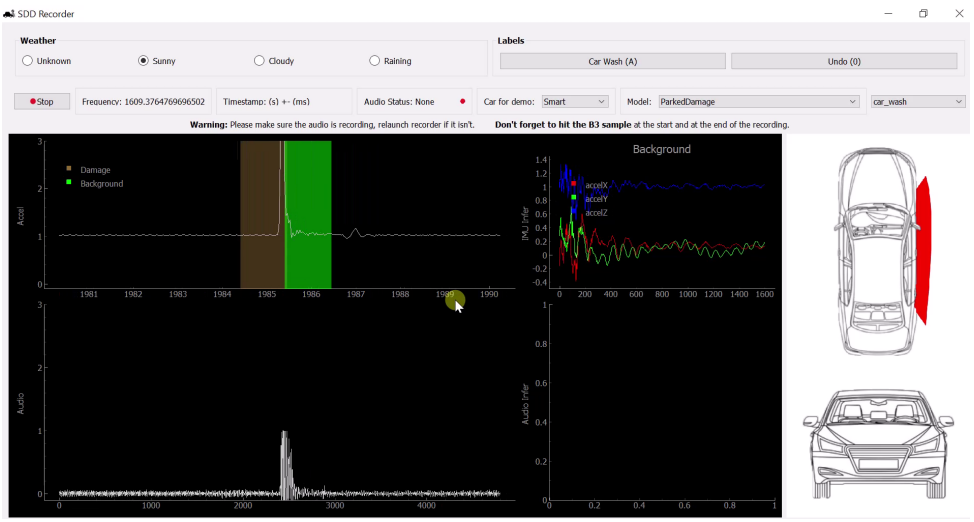


Figure 5.1: Recorder user interface

In the image above it is possible to observe multiple points of information. The main one taking the majority of the size window contains the signal retrieved with post processing happening in real time on the top part and on the lower part is shown the audio signal. In the image 5.1 it is also possible to observe the accelerometer signal with a brown and green highlight. Brown refers to a damage event classification and the green refers to a background event. Other data available to observation is the raw signal retrieved by the sensor and the possible car location of the impact inflicting event.

The second stage only happens after data has been collected. Verification is done to ensure if the labelling was done correctly, corresponding to the correct event and time interval. To perform this task, another software tool built by the main team enables the visualization of the signal obtained and the label associated with it. This procedure allows for corrections or a better interpretation on the event itself allowing for the post-processing to be as reliable as possible.

Figure 5.2: Labeller user interface

The image above is a representation of the data in the software tool, allowing for visualization of the accelerometer, gyroscope and audio signal. In the figure 5.2 multiple events are displayed with the associated label displayed in a box located on the right side. For each labelled event a deeper analysis can be made by selecting the label and zooming in to inspect the signal form and values.

Figure 5.3: Labelled event

For a designated event the time frame is delimited by the start and end time containing the relevant part inside of it.

5.2 Data preparation for visualization

Due to the scale of the data collection many components were not the same, mainly vehicles and sensor setups. This led to inconsistencies in the data making it impossible to analyse and interpret in a reliable and trustworthy manner. In order to rectify this, it was necessary to prepare the data so it was more consistent. Information contained in the respective JSON and HDF files was merged into a single HDF file.

5.2.1 Accelerometer rotation matrix

To achieve data equality the first step is to correct the rotation matrix according with the vehicle where the data is collected. The sensor is mounted to the top of the windshield near the rear-view mirror. This means by having the sensor in that location the angle of the windshield affects the data collected.

Applying the rotation matrix specific to the vehicle corrects this problem. The following image represents the vehicle coordinate system in place in accordance to the ISO 8855:2011 directive.

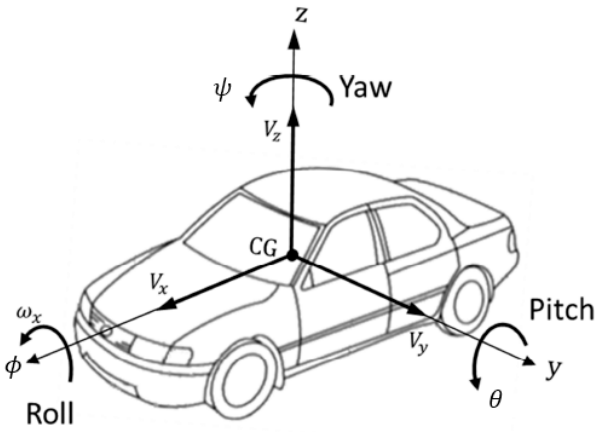


Figure 5.4: Vehicle coordinate system ISO 8855:2011

Transforming the rotation matrix impacts the accelerometer signal. Below is the signal before applying any processing. This signal is retrieved from a stationary run where the only event is opening and closing doors.

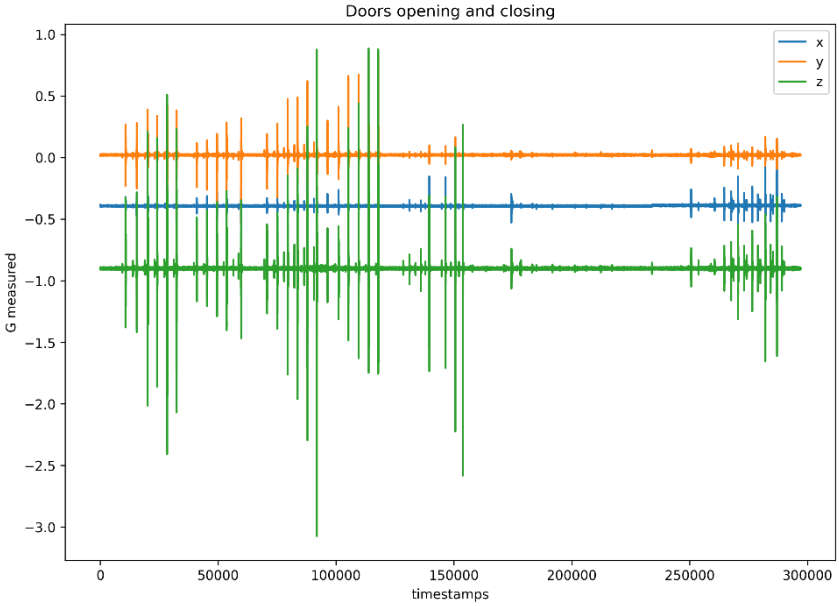


Figure 5.5: Original accelerometer signal

From the figure above it is possible to assume the following components: Axes are not aligned and the axes' values are not taking gravity forces into consideration. The rotation matrix fixes this misalignment and returns the signal in the following shape.

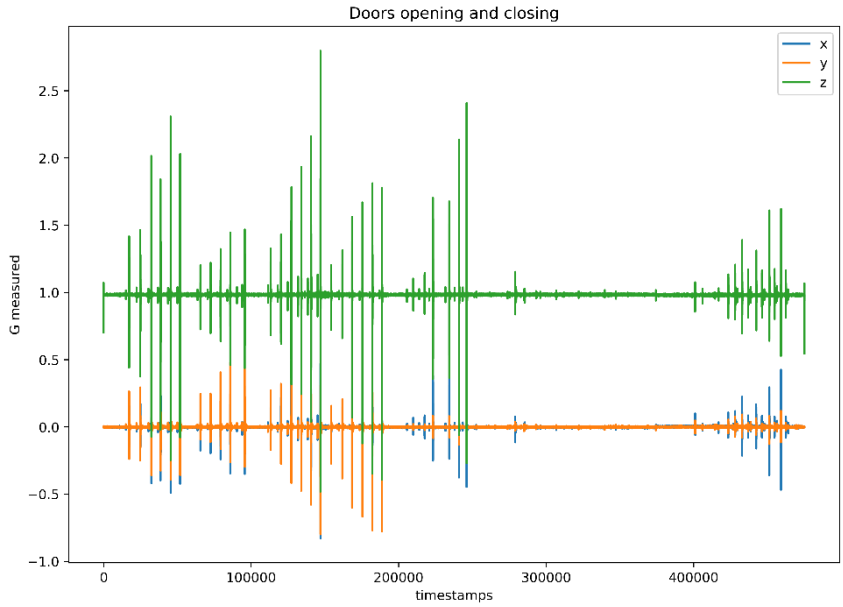


Figure 5.6: Processed accelerometer signal

Differences between the figure 5.5 and figure 5.6 are instantly noticeable. In the processed signal the x and y axes values remain at 0 when stationary as is it should since there is no acceleration or braking happening in a stationary position. Z-axis maintains a value of 1 when stationary to compensate for the acceleration caused by gravity.

All data was resampled to 1600HZ and a low pass filter was applied at 218HZ.

5.3 Exploratory Data Analysis

After collecting all data an analysis on it provides an in-depth overview of the features and conclusions that can be retrieved. This exploratory data analysis concerns only the data from the accelerometer, the data collected by other sensors is not considered for not being the key component of this thesis analysis. However, such data can be useful for further analysis and implementations to be made in the future.

In this chapter a statistical analysis on all the data is done but also a detailed analysis of a chosen set of events in order to interpret not only the signal but all the processing that can be done, thus obtaining extra information that can contribute to a better performance of the model to be created.

5.3.1 Folder statistics

The scope of this thesis is the detection of damage caused by an impact. To achieve this the model requires data to train and validate, which can be achieved by having two different folders for this purpose: One containing all data for training and another for testing if the learning was correct or not. Both folders differ in several components disfavouring in particular the testing folder, which in terms of size is considerably smaller and contains events with damage that do not appear in the training folder. All of this will have

impact from here forward, especially in the NN domain where a reduced number of information impacts the performance by not allowing generalization.

5.3.1.1 Train data folder

This folder contains two entries regarding data collection sites: Location A and location B. Each location provided data collected from multiple vehicles and test areas. Twenty nine hours are registered across 2795 files. The following graphs provide a discrete view of the quantitative composition of the data in this folder. For each graph a brief explanation of the numbers presented are given.

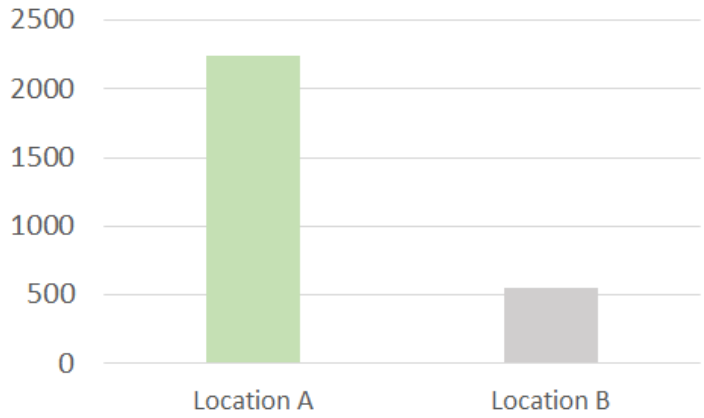


Figure 5.7: File Collection by Location Frequency Column Chart

From the available 2795 files, 2242 were collected at the location A while the remaining 553 were collected at the location B.

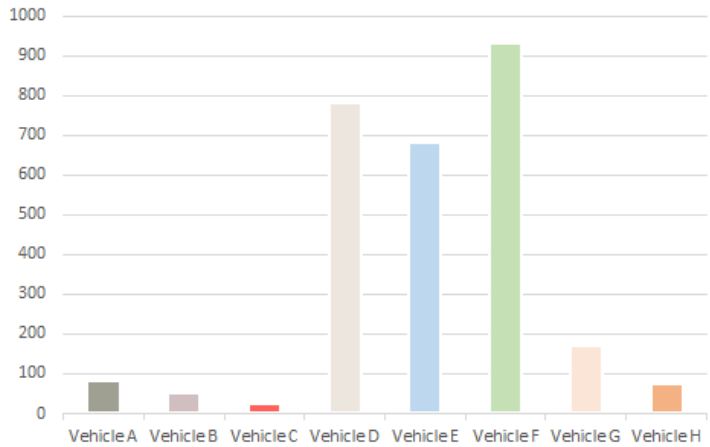


Figure 5.8: Vehicles Used in Collection Frequency Column Chart

From the graph above it is possible to count the number of different vehicles used, 8 in total, with the vehicle F being present in 930 files and the least present vehicle being only present in 23 files.

Vehicle D, E and F account for 2395 files that is equivalent 85% of the total number of files, displaying the unbalance regarding vehicles used for data collection. Values like this have a strong influence when classifying damages in vehicles with a low number of events collected.

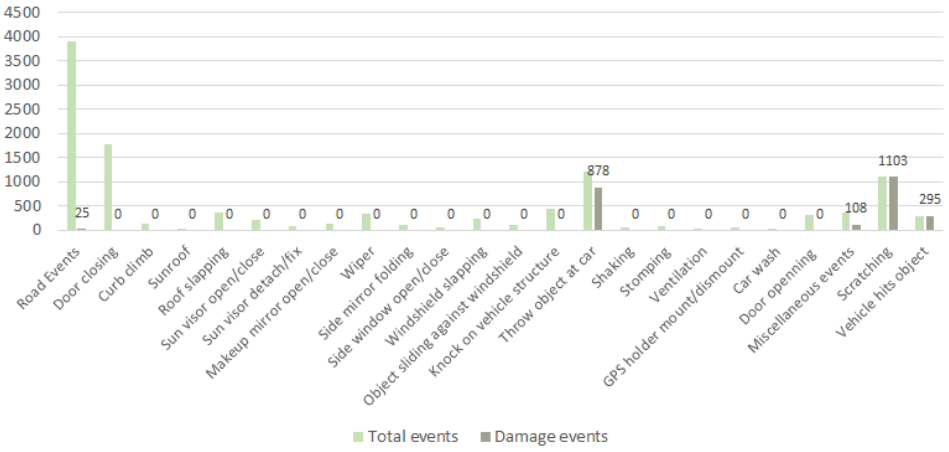


Figure 5.9: Events Frequency Column Chart

A total of 11405 events are available for modelling. However, only 2409, 21%, are reported with damage.

As previously noted, this unbalance degrades the performance of classifying algorithms by failing to provide the necessary information for a NN to learn from gross data.

The following events are associated with damage:

- Road events: 25 events reported with damage from a grand total of 3889.
- Throw object at car: 878 events reported with damage from a grand total of 1208.
- Miscellaneous events: 108 events reported with damage from a grand total of 362.
- Scratching: 1103 events reported with damage from a grand total of 1103.
- Vehicle hits object: 295 events reported with damage from a grand total of 297.

5.3.1.2 Test data events

This folder contains just one entry regarding location B as a collection site. Provided with data collected from multiple vehicles. Ten hours are registered across 2795 files.

After the gross statistical analysis of the training folder with records of the number of files, locations and vehicles used, it is now time to elaborate this analysis by creating a statistic of the events present in this folder. The following graph is composed by the name of the event along the x-axis and the frequency of events in total and events that have the damage status confirmed.

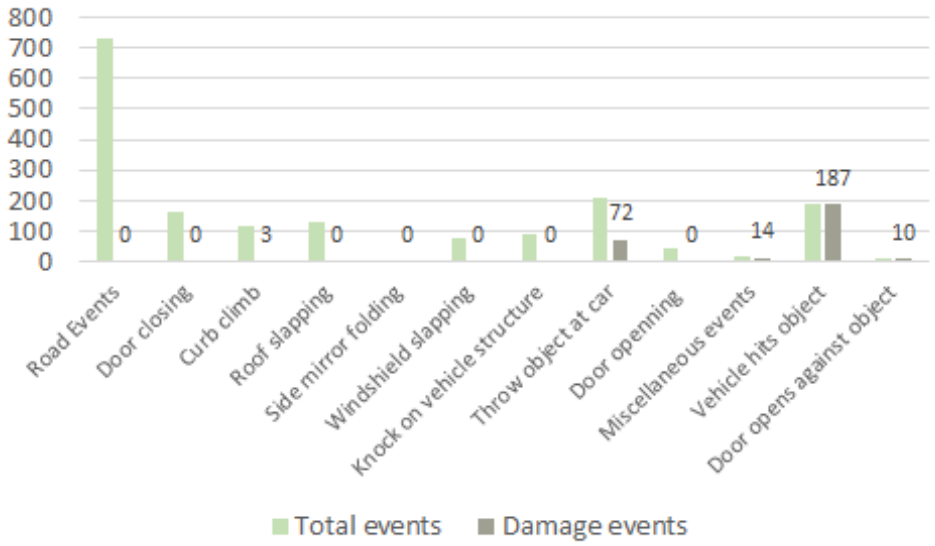


Figure 5.10: Events Frequency Column Chart

A total of 1781 events are labelled according to the associated event and from that number 286 are reported with damage only adding up to 16% of all events collected.

The following events are associated with damage:

- Curb climb: 3 events reported with damage from a grand total of 119.
- Throw object at car: 72 events reported with damage from a grand total of 208.
- Miscellaneous events: 14 events reported with damage from a grand total of 20.
- Vehicle hits object: 187 events reported with damage from a grand total of 189.
- Door opens against object: 10 events reported with damage from a grand total of 10.

Miscellaneous events is a set of events covering everything from speeding over bumps to emergency stops using ABS. This category also includes events collected in stationary or moving situations.

5.3.2 Event analysis

After the mention of events with damage and without damage, in this subsection a detailed analysis will be dedicated to each type of event with special focus on several components. The purpose is to demonstrate that through the accelerometer signal it is possible to detect the differences between events not only on a first view analysis but also through processing.

Through the use of FFT it is possible to transform the signal from a time representation to a frequency representation.

Once the frequency and temporal domain is observed, one way to be able to further analyse and explore what the signal indicates can be through wavelets. Throughout the study of the accelerometer data it was defined that values below a frequency of 10 Hz are considered as noise.

With the ability to decompose an original signal from a time and frequency domain perspective in a discrete or continuous manner it is possible to observe at different frequency and time scales the signal intensity. This transform is extremely effective in signal processing.

Through wavelet analysis it is possible to extract information about the presence and intensity of the signal present across all scales.

Prior to all this analysis, the euclidean norm of each point was calculated on the accelerometer axes to analyse the data points as just one set of data and not 3 axes. This processing consists of transforming the original data signal as represented in the figure 5.11 into just one set of data points.

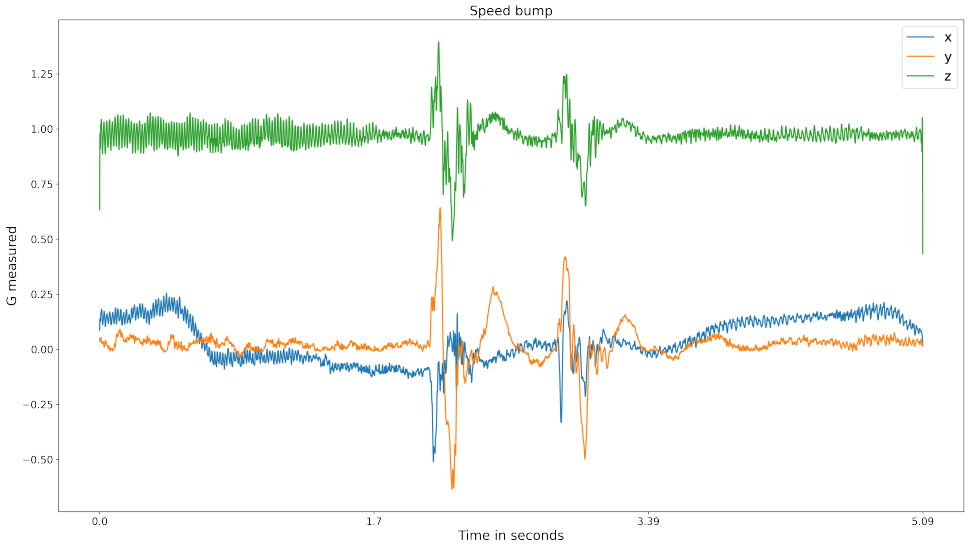


Figure 5.11: Accelerometer signal represented by the X,Y and Z axis plot

Taking advantage of the vector norm equation, $v = \sqrt{x_1^2 + x_2^2 + ... + x_N^2}$, adjusting this equation to the accelerometer signal and its axes returns the following equation:

$$v = \sqrt{X^2 + Y^2 + Z^2}$$

Where for each point in the event, it is calculated the norm of that exact time point in the event allowing the creation of a single set of data points.

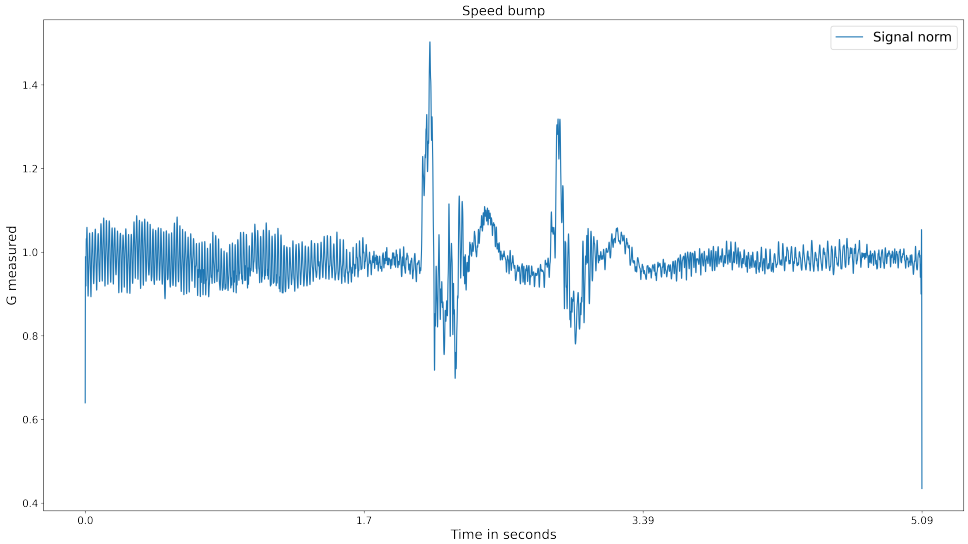


Figure 5.12: Accelerometer signal vector norm plot

The resulting plot displayed in the figure 5.12 allows getting the magnitude of the accelerometer force that characterises the event which is the essence of this transformation. This step is extremely important for all processing and analysis that follows.

5.3.2.1 Non-damage event

For non-damage events, actions such as closing doors reveal the presence of associated high energy while not being an impact or causing damage. For a further in-depth analysis of non-damage events it is necessary to look at other common examples but this time in moving situations. The next event serves as comparison between a moving and stationary event.

Starting with a moving non damage event, in this case a speed bump passing. The figure 5.13 displays the accelerometer signal plot during the time frame.

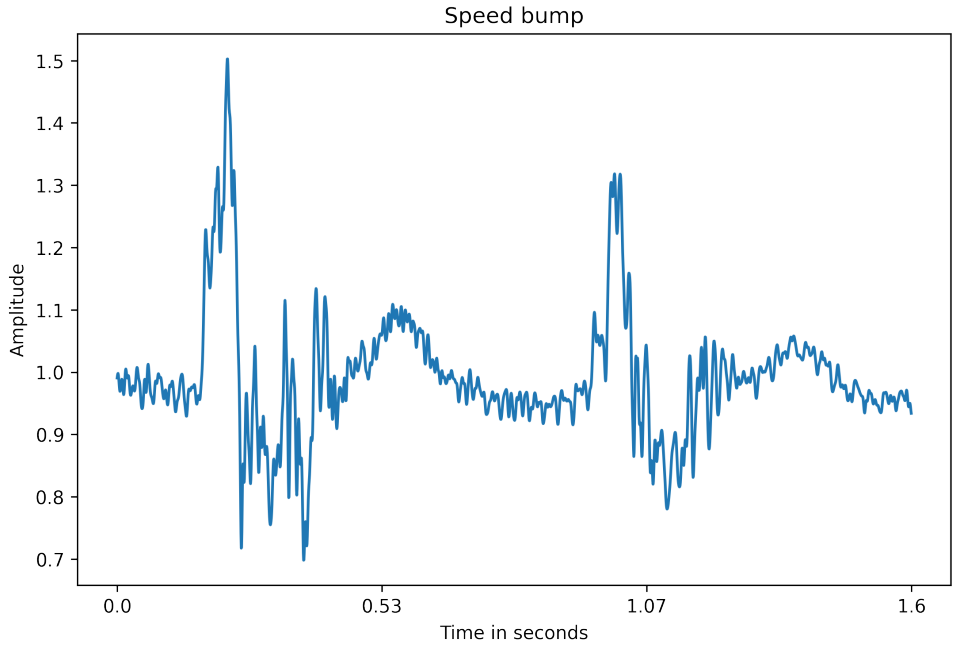


Figure 5.13: Speed bump signal event

From the signal is possible to analyse two peaks referring to each vehicle axle passing through the bump and the intensity expressed in acceleration values. The duration of the signal also represents a standard situation in non-damage events which is a long duration time event.

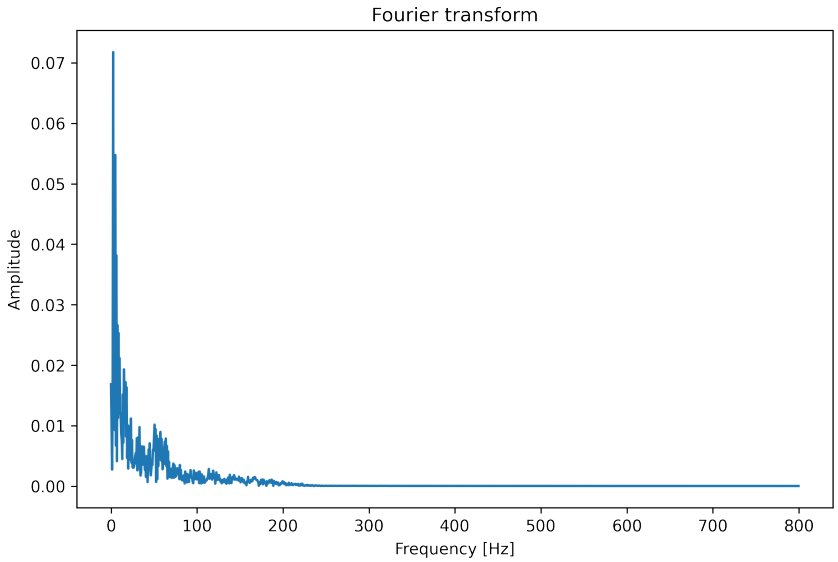


Figure 5.14: Speed bump event FFT

Low frequencies are presented with higher amplitudes when compared to higher frequencies, expressly in the range of 10-50Hz. It is also a standard situation in non-damage events, rarely exciting higher frequencies.

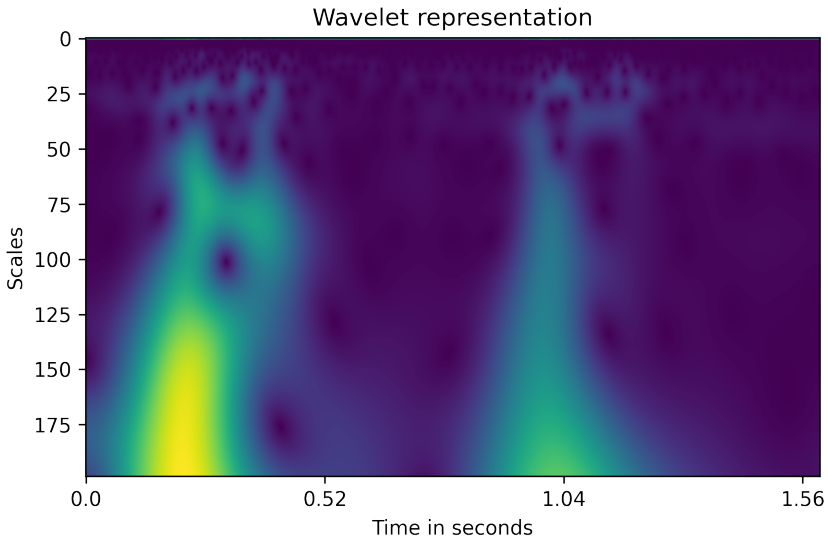


Figure 5.15: Speed bump event wavelet

The figure 5.15 allows for a wavelet representation with the scale and intensity components. Brighter colours represent a high intensity and darker colour representing a lower intensity. The beginning of the event has a strong intensity at the top end of the scale.

Wavelets can also be useful to decompose the signal in time and frequency.

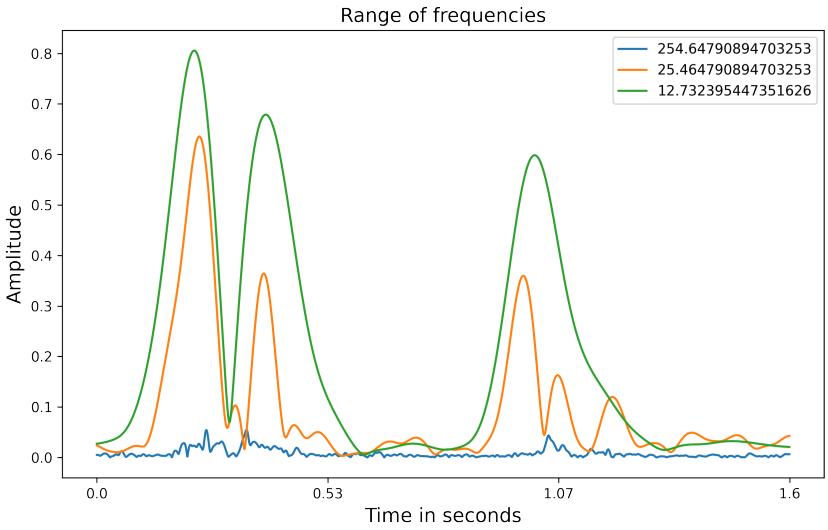


Figure 5.16: Speed bump frequency decomposition wavelet event

Another non-damage event in analysis is a simple door closing event, which despite being a normal procedure when entering or exiting a vehicle involves a piece hitting the structure of the vehicle. This event behaves very closely to a damage event derived from the forces involved.

This behaviour is similar on all doors but since the sensor location is on the windshield, the front doors cause a greater impact. With this observation in mind the following door closing event is based on the front passenger.

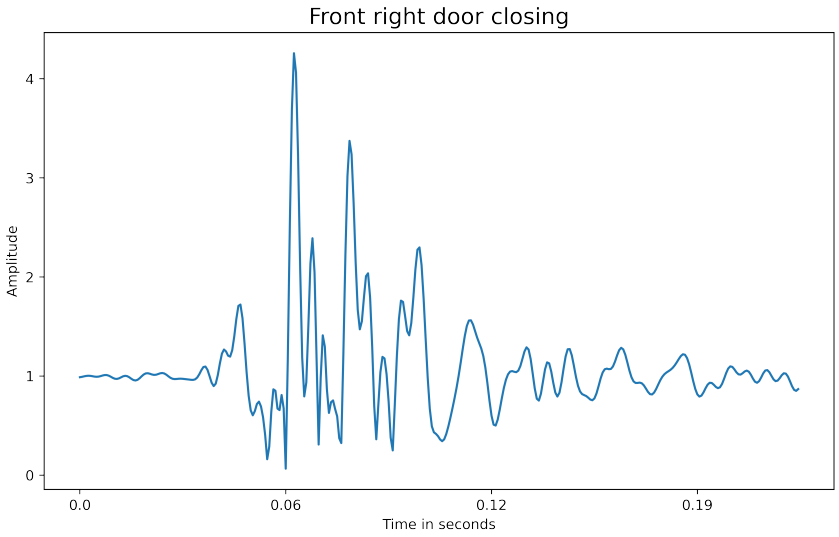


Figure 5.17: Front right door closing signal event

From the y-axis values it is possible to see the impact generated is a little over 4 G's, which for a non-damage event is extremely high, acting as an outlier compared to the other events. But as mentioned before, the duration is extremely short, ± 0.06 seconds. This event does not end in the time frequency analysis, when transforming the signal into a frequency representation some relevant observations can be visualized.

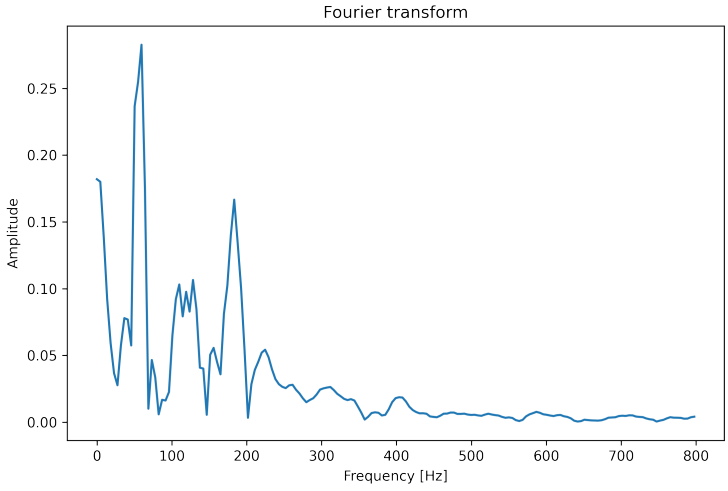


Figure 5.18: Front right door closing event FFT

As observed from the figure 5.18 all frequencies up to the low filter limit at 218Hz are present with a high amplitude. Around the 50Hz mark the presence is noticeable as previously seen. This time at the range from 100-200Hz there is a considerable amount of amplitude value.

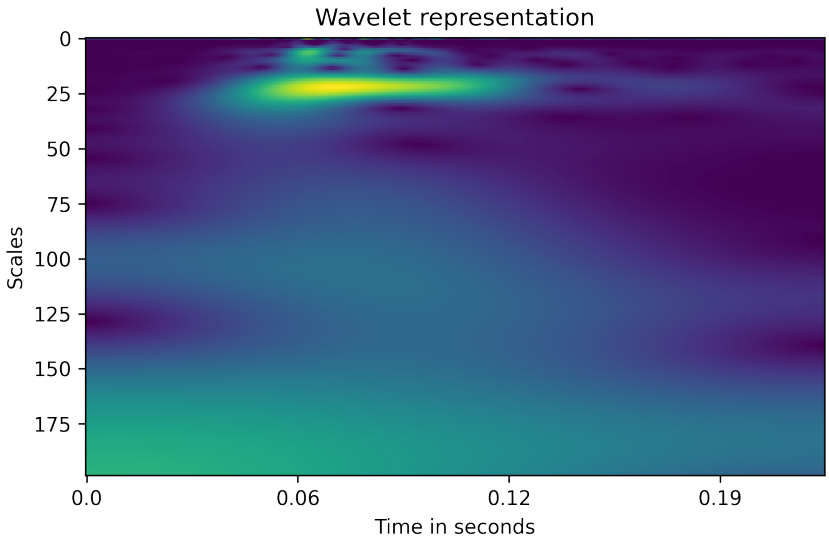


Figure 5.19: Front right door closing event wavelet

Contrary to the results presented in the figure 5.15 where across the scale shows up as independent clusters at different scale levels and not as a continuous line across the plot. The energy is also dispersed all around the time-frame and not at the ends of the plot as in the speed bump event. The figure 5.20, decomposes the frequency into ranges: Low, Medium and High. The latter remains untouched during the event, this is characteristic in non-damage events.

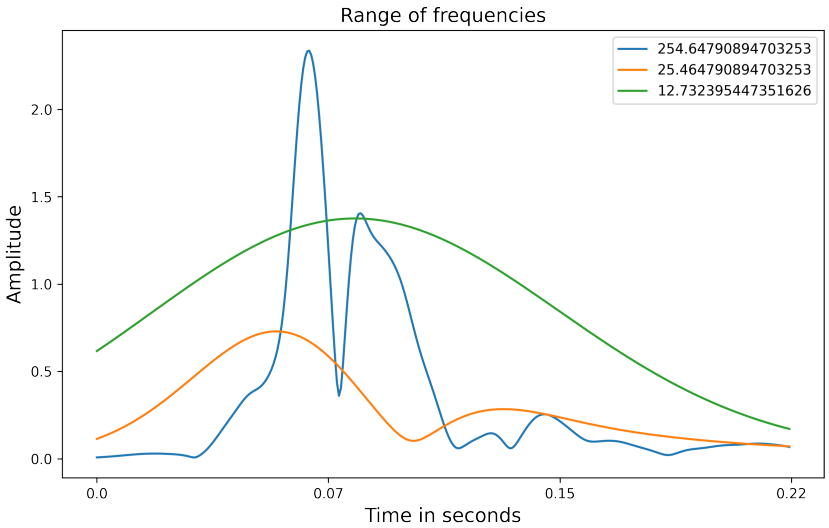


Figure 5.20: Front right door frequency decomposition wavelet event

The figure 5.20 despite belonging to a non damage event contains values different to those observed in the figure 5.16. Frequencies in the high range are present in two peaks containing a high value in terms of amplitude.

This reinforces the previously mentioned statement, doors when closing transmit high energy impact into the vehicle structure. The presence of frequencies in the medium range are also present with a relevant

amplitude and finally the frequencies in a lower range reach a peak 5 times lower when compared to high frequencies.

5.3.2.2 Damage event

Once non-damage events were analysed and visualised the same procedure was applied to events that inflict damage as a final result. All the following events have been acquired for the sole purpose of understanding the components of an impact that consequently generates damage.

Just by analysing the accelerometer it is hard to discern between door closing events and damage events. Both produce a similar signal with high frequencies associated, however, door events have a higher time duration.

Starting with a vehicle hits an object with its front bumper and rear bumper. This event contains a signal with a high amplitude due to the fact the vehicle becomes stationary when it hits the object, this sudden stop releases a great amount of energy.

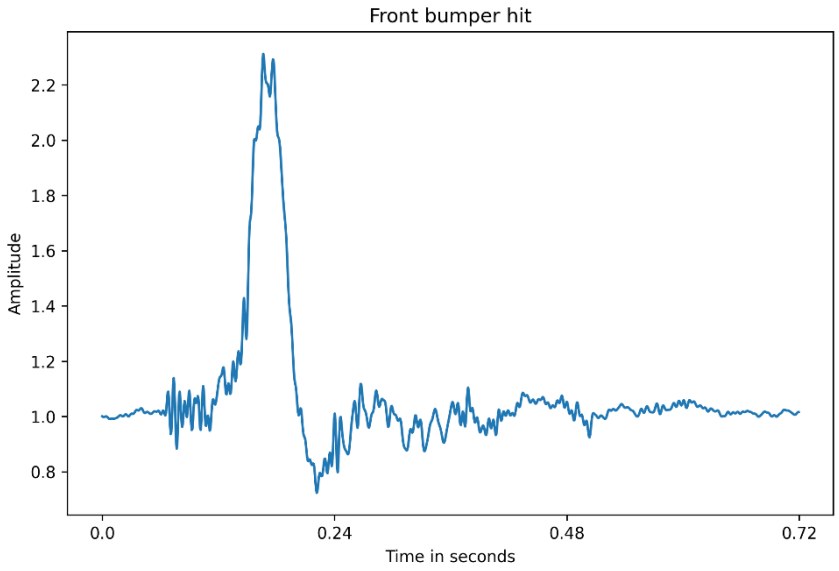


Figure 5.21: Front bumper hits object signal event

The impact generated a force with a value of 2.2 G's and duration of ± 0.06 seconds, reflecting a typical impact action with high energy and low time duration.

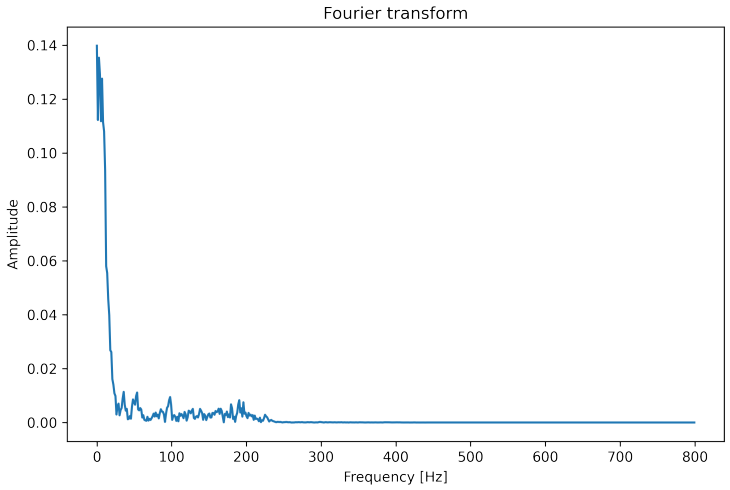


Figure 5.22: Front bumper hits object event FFT

Despite the frequencies in the range of 0-218Hz containing positive values in the amplitude axis, the peak is located in the lower frequencies near 0Hz where the amplitude reaches the maximum value with the amplitude decreasing throughout the rest of the range, this behaviour contrasts with a classic damage event.

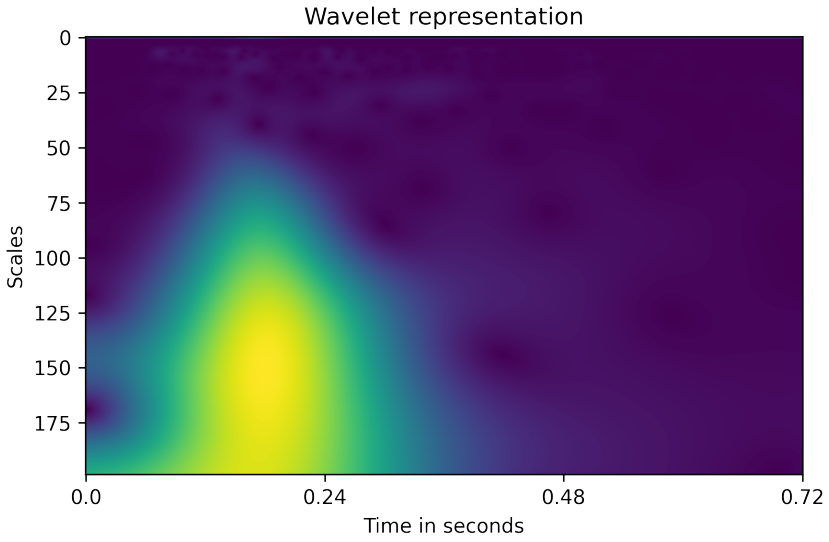


Figure 5.23: Front bumper hits object event wavelet

Analysing the wavelet plot it is possible to recognize the location of the event in the timeline, located with a significant energy intensity between the 0.14 and 0.29 seconds mark. Also noticeable is the existence of such high intensity on the higher scale range leaving the lower part of the scale untouched.

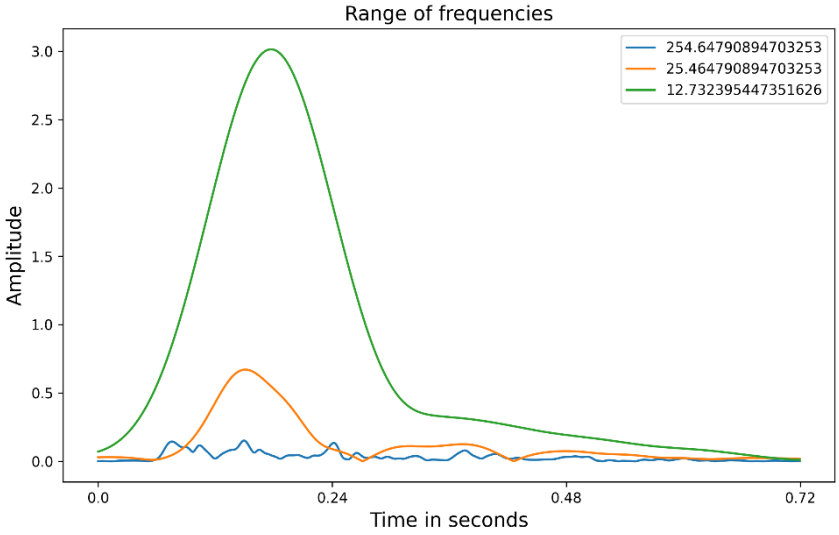


Figure 5.24: Front bumper hits object frequency decomposition wavelet event

There are a number of similarities between the figures 5.22 and 5.24, both showing peculiar results compared to the expected behaviour of a typical damage event. Lower frequencies overshadow other frequencies by an extensive margin.

In the case of a rear bumper hits object event, there were some similarities found between the signal behaviour and a front bumper hits object event. The figure 5.25 exhibits the peak value at the moment of impact and the time duration. Both components identify a normal damage event.

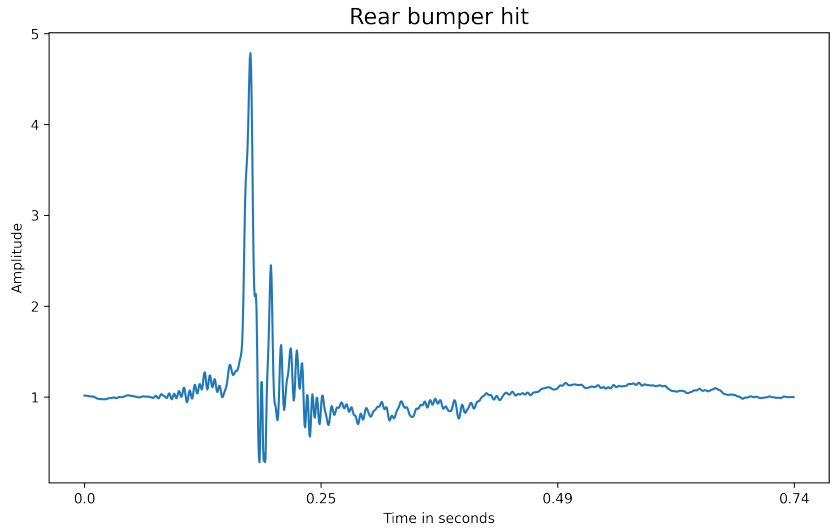


Figure 5.25: Rear bumper hits object signal event

Signal evaluation using the FFT displays similarities to the ”front bumper hits object”event signal FFT displayed in the figure 5.22. Lower frequencies are present with a higher amplitude, specially in the 0-50Hz range and beyond this similarity, the frequencies above 100Hz display an amplitude value higher in comparison with the figure 5.22.

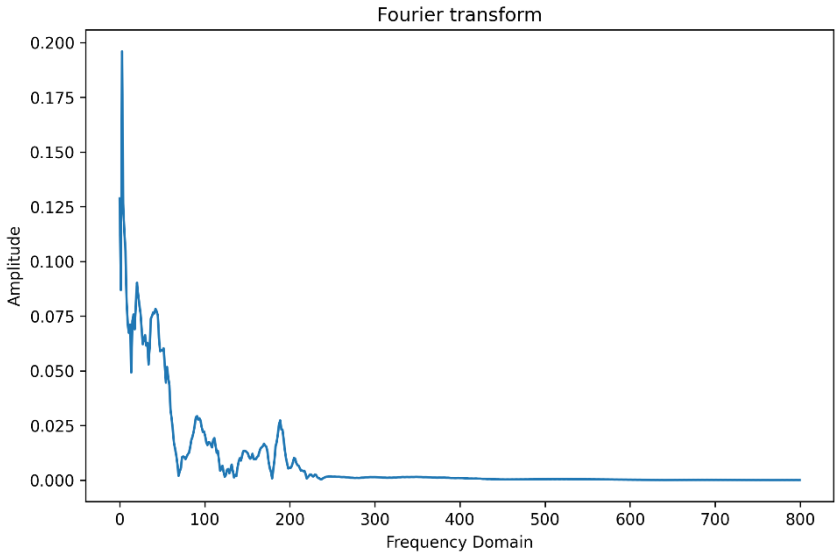


Figure 5.26: Rear bumper hits object event FFT

In comparison with the wavelet plot displayed in the figure 5.23 there is a dissimilarity with the scale range. The signal in the figure 5.27 is present across the scale whereas the signal in the figure 5.23 only covers the high end part of the scale

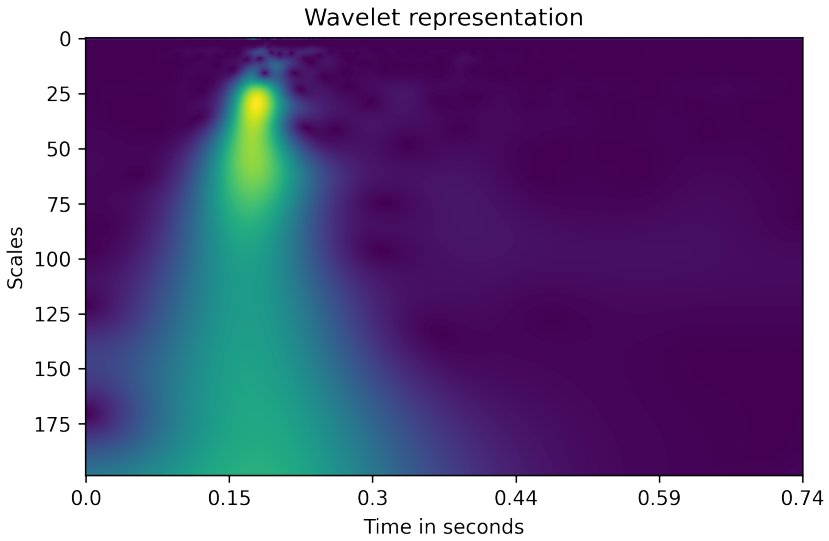


Figure 5.27: Rear bumper hits object event wavelet

In the figure 5.28 once more the signal decomposition in frequencies displays the presence of all ranges, however the high frequency range is not the predominant one. Frequencies in the medium range are the most predominant one followed up by frequencies in the lower range.

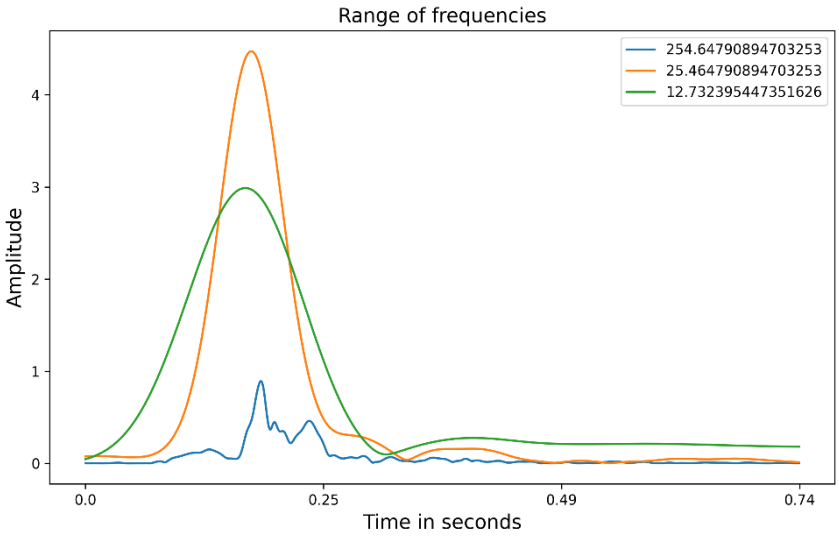


Figure 5.28: Rear bumper hits object frequency decomposition wavelet event

For the last damage event analysis, door opening against an object is the use case to compare against the non-damage event, front right door closing. Some significant aspects should be acknowledged beforehand, the non-damage event considers a door being slammed with excessive force which results in a high load of energy being transferred to the vehicle chassis leading to an unusual behaviour among non-damage events.

For this event, the door is open against an object, transferring the energy into that object which is not the vehicle and the damage created is the contact point between the door and the object itself.

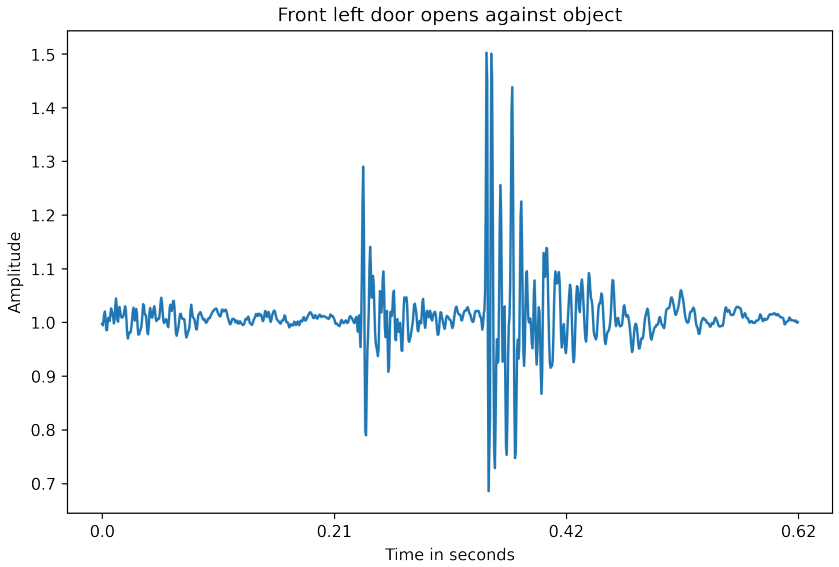


Figure 5.29: Front left door opens against object signal event

In comparison with the values reached in the figure 5.17 it is possible to observe the large difference between the events, however, this event produces a a high G-force value in a short period of time corresponding to a damage event.

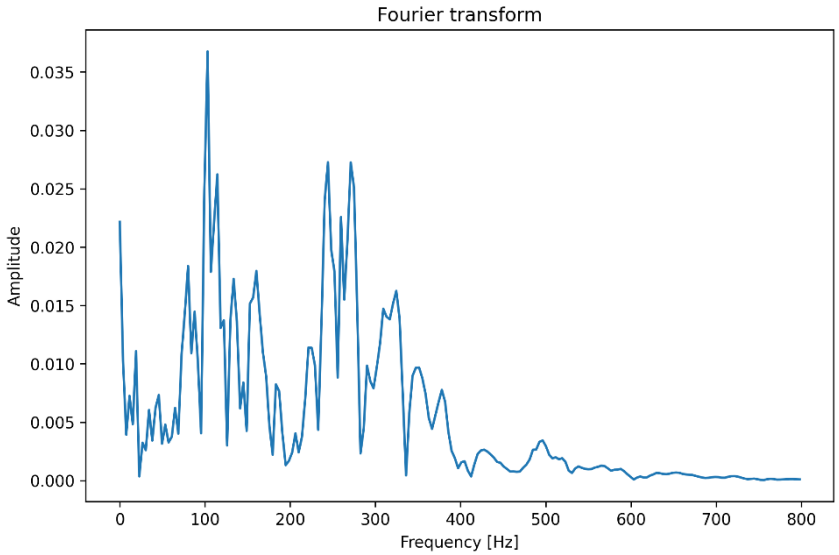


Figure 5.30: Front left door opens against object event FFT

All frequencies in the range of 0-218Hz contain positive values, frequencies above the 218 Hz limit cap also display positive values with the range between 200-300Hz reaching values near the peak amplitude frequency, 100 Hz. This finding, not seen until now, reinforces the difference between damage and non-damage events.

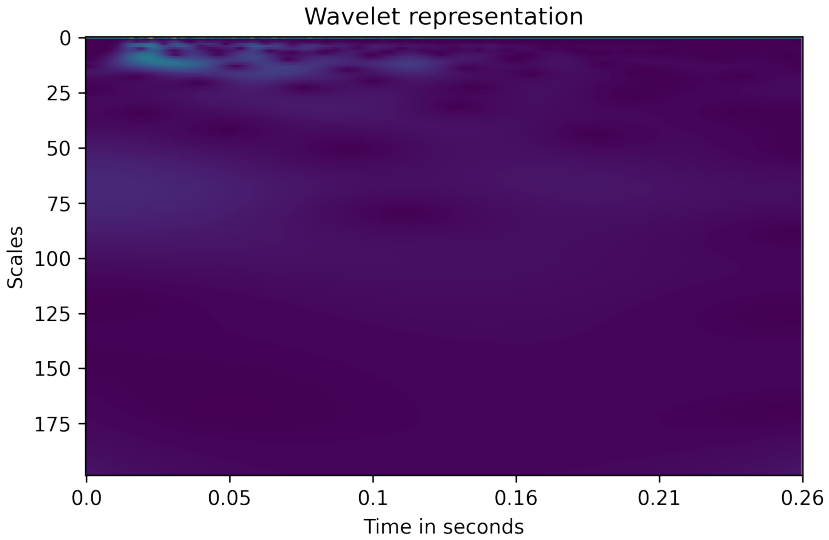


Figure 5.31: Front left door opens against object event wavelet

Contrary to expectations, the wavelet plot does not contain areas of high intensity. At a lower scale level it is possible to observe some different intensities however they are not as striking as previously seen in figures 5.27 and 5.23 where the event was of a non-damage type.

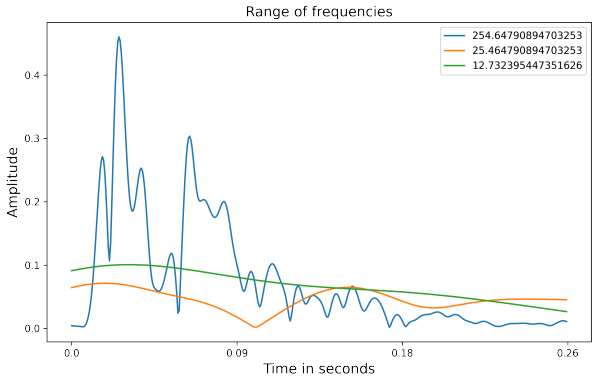


Figure 5.32: Front left door opens against object frequency decomposition wavelet event

Unsurprisingly, higher frequencies are present with a greater significance compared to the other frequency ranges, which behave in a similar way across the time range. In addition to this observation, it is also possible to monitor the decay in amplitude after the damage event, an abrupt decay is a key feature for detection of damage events.

5.3.2.3 Damage and non-damage event comparison conclusion

Taken together the findings of this analysis support the idea that it is feasible to isolate damage and non-damage events based on components identified in the signal. Once all cases were analysed using the same procedure, the results showed there are differences in every component analysed. For the signal itself, the G-force value is higher in a damage event compared to a non damage event. However, as previously outlined, door closing act as an outlier among non-damage events with acceleration values comparable to damage events.

Applying a FFT to the signal plot, returns a plot of the peak amplitude values for all the frequencies showing that damage causing events contain higher frequencies when compared to non damaging events. Frequencies at lower values, 0-50 Hz, are similar in all events, with discrepancies between the events being apparent only in frequencies above 100 Hz.

Wavelet analysis reveals the intensity of the signal and the level of scales affected. The figure 5.32 when compared to the rest of damage events presented in the figures: 5.23 and 5.27. However, when compared to the door event presented in the figure 5.19, the appearance is related at a lower scale number.

Complementary to the wavelet plot, deconstructing the frequencies and respective amplitude throughout the time window allows to see the range of frequencies relevant for identifying the type of event. On damage events the presence of high frequencies is distinguishable from the other range, whereas in non-damage events the presence of high frequencies is not as common, with frequencies in the lower ranges being more dominant.

Generally speaking, the analysis made on both types of events, highlighted the characteristics and the discrepancies of both events.

5.4 Data preparation and selection

Prior to the modelling phase, the data is not prepared to be ingested by the algorithms. Achieving this, involves selecting what data is considered an event, either damage or non damage. The data is already labelled which helps to simplify the process. However, there are other factors that do not share the same pre-processing.

Selecting what data to use is the first approach to be executed. The process described in the section 5.3.2 is applied to the data available. An euclidean norm is applied to the 3 axes in each point in order to access the magnitude generated by all three axes: X, Y, and Z.

To decide what data is selected to be used as input in the modelling process, there was a need to filter all relevant data. To enable this, the study previously done in the section 5.3 gave an overview on all signal characteristics. For this case, the analysis on the accelerometer values is extremely relevant for the data selection.

For all labelled events: Damage or non-damage, the maximum labelled acceleration values were submitted to a comparison to visualize similarities. Moreover, the following plot displayed in the figure 5.33 presents the cumulative distribution function for the acceleration values regarding damage and non-damage events.

For acceleration values between the range of 1 and 1.5 G-force, the split between both lines is clear and obvious. The plot on the right side presents the cumulative distribution function for both type of events. The abrupt peaks at the acceleration value of 1.0 should be ignored due the abrupt changes in distribution.

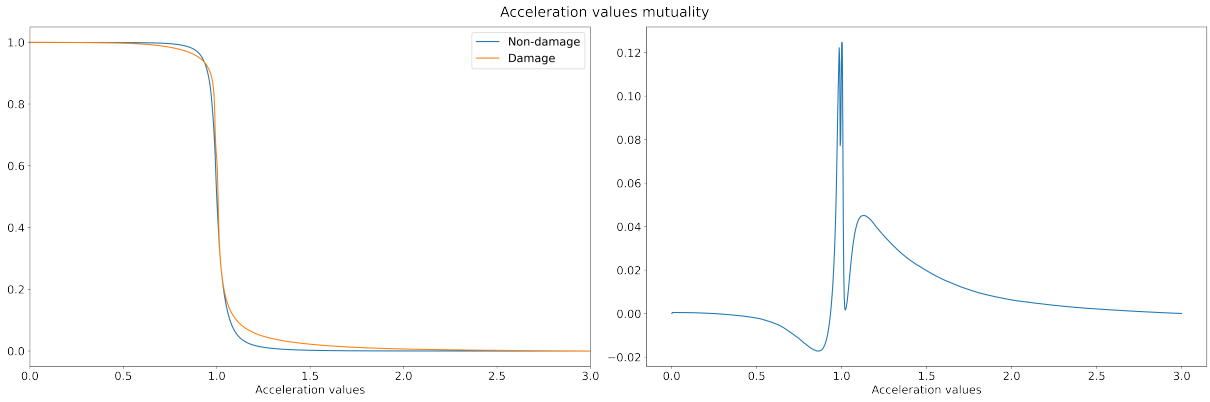


Figure 5.33: Cumulative distribution function comparison of acceleration values for damage and non damage events

It was found that the acceleration values after the 1.0 value starts showing differences. The plot on the right side in the figure 5.33 confirms that the best threshold for the acceleration value starts around the value of 1.1. Any value above that is an acceptable choice for setting it as threshold. However, the higher the threshold value the less events are available for selection. Therefore, the best acceleration value for threshold should be as small as possible to allow for the selection of a larger number of events.

The acceleration value of 1.1 corresponds to this requirement. However, the main team uses the value of 1.2 as threshold value for selecting events. To be consistent with the main team the acceleration value

of 1.2 was also selected as threshold for selecting events in the algorithm.

Once the threshold value was discovered, data selection remains incomplete, as there are two different datasets to consider: Train and testing datasets.

For both datasets the data pre-processing is equal for all files. The process presented in the subsection 5.3.2 is applied. Afterwards, depending on what dataset is being worked on, the selection method is different according to the requirements of the algorithm.

The selection process starts by identifying the points where the threshold of 1.2 is exceeded that point is highlighted as the starting point for the following steps by creating a window capable of containing the event.

The following indexes are used for the window timestamp:

- Window start index: From the first index of the labelled event where the threshold value of 1.2 is exceeded, 160 points are subtracted, 10% of the sampling frequency of 1600 Hz.
- Window end index: From the first index of the labelled event where the threshold value of 1.2 is exceeded, 1440 points are added, 90% of the sampling frequency of 1600 Hz.

5.4.0.1 Scratch events

The restricted use of events that could be labelled as damage account for the exclusion of other events that were labelled as damage since the beginning.

Scratch events fall in this category, reasons for excluding this event are complex but justified. Scratches cause damage, like a key scratching a body panel or any other part. However, since the focus of this thesis is using exclusively the signal retrieved by the accelerometer, the data is not sufficient to distinguish between damage or not. Initial analysis on the signal did not give explicit information that could be used.

Inevitably this event had to be excluded as the constraints of the project the ability to detect and classify damage done by scratches was left rather limited.

There are other possibilities to address this issue such as multi sensor data fusion, combining accelerometer and microphone data can be a solution to approach this issue. The additional information provided by the microphone may be the missing factor.

It can thus be suggested there are ways to detect scratches. Unfortunately this thesis in particular does not deliver the capacity to perform the scratch event classification.

Chapter 6

Modelling and Evaluation

With the completion of data analysis, interpretation, preparation, and selection, the next step in the ML cycle is modelling and evaluation.

A considerable amount of time is spent making sure that every detail is correct at this iteration in order to achieve reliable results.

This sets the stage to introduce the algorithms implemented to model the data and to accomplish the correct detection and classification of impacts that cause damage. Since the focus of this thesis is deep learning, all algorithms are embedded in this domain. Within the framework of this criteria, there was space to explore the various NNs.

There is a large amount of NN to explore in the context of this thesis. However, due to a variety of constraints, the development was limited to what was considered the most appropriate algorithms for this use case.

Research made in the chapter 2, revealed vital information related to the algorithms to implement and allowing for the development being done in the time frame that was available for the thesis. Since this was under the domain of deep learning with supervised learning, the starting point was the development of a MLP algorithm, followed by a CNN algorithm and finally a RNN algorithm.

For all algorithms the implementation passes through the data ingestion, transformation, modelling, and benchmark.

All models used the train data as input, where for the validation dataset it was used 10-fold cross validation. For evaluation the test data is used to benchmark if the model trained correlates to the test set.

Benchmark on models was carried out in a computer with the following specifications.

CPU	Intel Core (TM) i7-7800X
CPU clock speed	3.50GHz
RAM	128GB
GPU	3x Nvidia GTX 1080 Ti 11GB

Table 6.1: Computer specification used for modelling and evaluation benchmark

6.1 MLP

For the MLP implementation the data ingested by the algorithm was different from the CNN and RNN. Instead of using the 1 second window containing the x,y and z axes and vector norms from these 3 axes another approach was used. Feature engineering was applied in order to extract and use the information obtained from the retrieved data.

From the initial data axis, the euclidean norm of the combination it is applied to create the axes: xyz, xy, zx and zy to increment the available information. From the initial 3 axes: x, y and z the incremented result from applying the euclidean norm results in having 7 axes: x, y, z, xyz, xy, zx and zy. With 7 different data points as the starting point to calculate and extract features, the next step was extracting those features. The review made in the chapter 2 helped in selecting features and with the usage of python libraries TSFEL[1] and TSFRESH[2]. Combining all sources of knowledge, a range of features was selected from various domains, mainly statistical, and time and frequency. After extracting and selecting the features to use as input data to the MLP network, the next step is transforming the vectors and vector norms data into the processed data. From the initial number of 7 axes for each event with duration of one second, feature extraction is applied to each axis. Wavelets are decomposed in 6 scales: 8, 16, 32, 64 , 128 and 256.

After all the feature extraction process is completed the final number of attributes is 1864 for each event.

6.2 CNN

For the CNN implementation the data ingested by the algorithm is based on the original data retrieved by the accelerometer with the adddition of the vector norms based on the 3 original axes. The input data is contained in an one second window containing the x,y and z axis and vector norms from these 3 axis resulting in the following axis: xyz, xy, zx and zy. The entry data is made up of seven arrays containing 1600 values representing a window of one second. The dimension choosen for the CNN algorithm was one dimension as this is the recommended approach for a time series due to the fact the kernel slides along the time frame. The CNN developed incorporates a kernel of size 3, sliding through the values of the 7 axes(x, y, z, xyz, xy, zx and zy), 3 at the time.

[1]Time Series Feature Extraction Library (TSFEL for short) is a Python package for feature extraction on time series data.

[2]tsfresh is a python package that automatically calculates a large number of time series features. Furthermore, the package contains methods to evaluate the explaining power and importance of those features for regression or classification tasks.

6.3 RNN

During the research made on the chapter 2, specially, the in depth analysis of driving behaviour we found a great number of articles using RNN algorithms to evaluate and classify what type of driving behaviour was involved. Furthermore, the most recent articles in all categories studied in the chapter 2 were related to RNN algorithms. The reason for this was the fast development and state of art in performance, and as such all this outlines a path to explore and this thesis is no different from that perspective. To emphasize the advances with these algorithms, three types of RNN were explored and implemented.

From a base RNN passing through an LSTM and GRU we explore the multiple approaches each algorithm provides according to the input data and classification performance. To simplify the process and to compare the performance in an equal manner, the structure of the algorithm is equal for the three approaches. The only difference is the layer type, which is in accordance to the type of RNN being implemented. The input data is equal to the CNN developed: 7 axes(x, y, z, xyz, xy, zx and zy) containing 1600 points representing a window with a time duration of one second.

6.3.0.1 RNN

For a simple RNN implementation, 4 layers were used. The input layer with all the data information, followed by 3 RNN hidden layers with dimension of 150 and ReLU activation. Then a linear layer before the output layer with dimension of 1.

6.3.0.2 LSTM

For a simple LSTM implementation, 5 layers were used. The input layer with all the data information, followed by 3 LSTM hidden layers with dimension of 100 followed by a dropout layer and then a linear layer before the output layer with dimension of 1.

6.3.0.3 GRU

For a simple GRU implementation, 4 layers were used. The input layer with all the data information, followed by 3 GRU hidden layers with dimension of 150 followed by a linear layer before the output layer with dimension of 1.

6.4 Evaluation

Along the state-of-the-art review many authors published results related to their work: 2.5, 2.7, and 2.10. However, during the review of the state-of-the-art the only metrics taken into consideration were accuracy and F1-score. Those were the only metrics presented in the majority of the articles reviewed.

When the EDA was concluded multiple concerns related to metrics appeared. The presence of an unbalanced in the number of events showed that using accuracy as the principal metric would not work

under this condition. Futhermore, the test dataset contains 9774 events with 9508 labelled as non-damage events and the remaining 266 labelled as damage events. Assuming the prediction to all events being labelled as damage the resulting accuracy value is above 90%.

In order to correctly evaluate the performance of an algorithm, another evaluation metric must be used instead of the traditional ones. To help with this task, the main team reported the metric they used when evaluating algorithms for this specific case where the data is unbalanced. The following is the mathematical formula for Mathews Correlation Coefficient (MCC) recommended by the main team.

$$MCC = \frac{(TP * TN - FP * FN)}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

The result is in a range -1 and +1, where:

- **-1** : Misclassification
- **0** : Random prediction (coin tossing classifier)
- **+1** : Perfect classification

In addition to the MCC another metric used was the confusion matrix that takes into account the number of FP. This is a client requirement, the lower the number of FP the better the model. The reason for this is based on every time an event is inferred as an impact event resulting in damage the vehicle must be analysed quickly as possible making it unavailable for hire. Costs in this case increase by a respectful value by removing a vehicle from the fleet for analysis and having an employee to assess the vehicle status. Avoiding this as much as possible is a requirement.

To test and evaluate the models implemented the following steps were done:

- Train dataset: Data retrieved from the train dataset, see 5.3.1.1.
- Validation dataset: 10 fold cross validation.
- Test dataset: Data used to evaluate the model with unseen data during training for the purpose of seeing if is capable in generalizing with unseen data.

More details about the results obtained and the hyperparameters used can be seen in the following subsections.

6.4.1 MLP

Hyperparameters		
	Learning rate	1e-4
	Batch size	256
	Epochs	75
Metrics		
	True positive rate	82%
	Validation MCC	0.93
	Test MCC	0.76

Table 6.2: MLP hyperparameters used and best result achieved

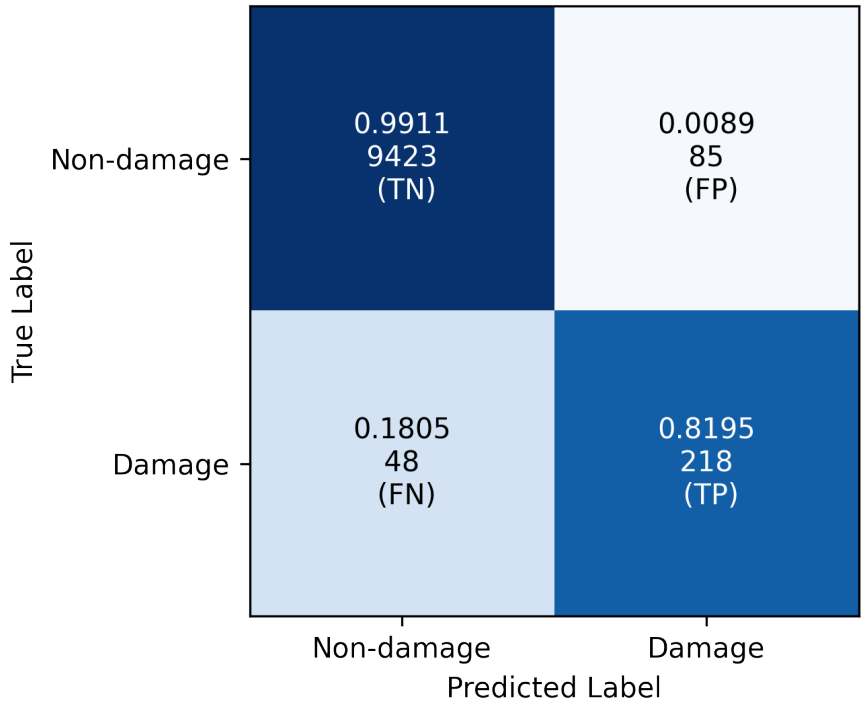


Figure 6.1: MLP confusion matrix test dataset result

6.4.2 CNN

Hyperparameters		
	Learning rate	1e-4
	Batch size	256
	Epochs	25
Metrics		
	True positive rate	66%
	Validation MCC	0.76
	Test MCC	0.74

Table 6.3: CNN hyperparameters used and best result achieved

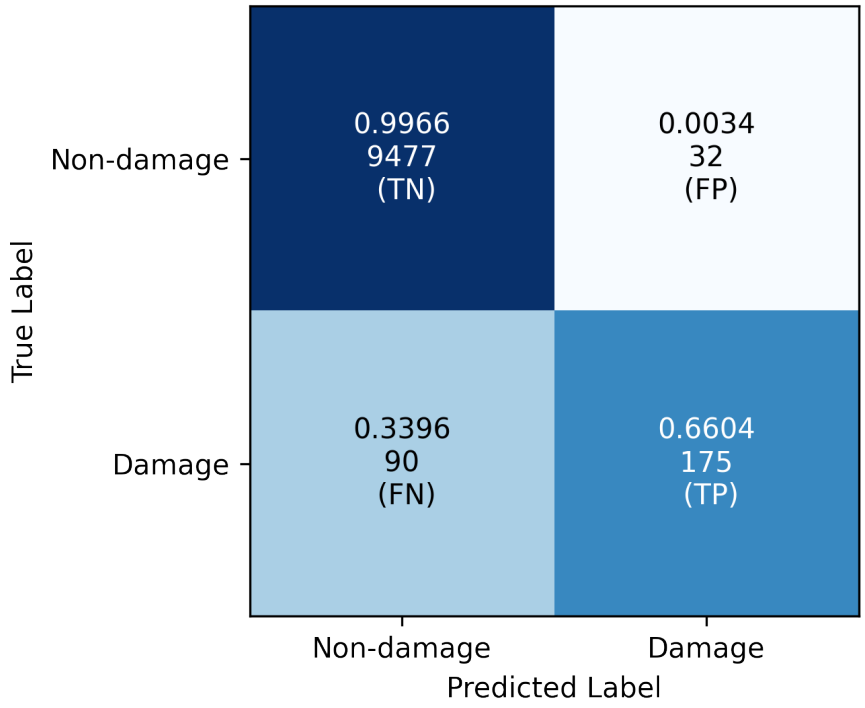


Figure 6.2: CNN confusion matrix test dataset result

6.4.3 RNN

6.4.3.1 RNN

Hyperparameters		
	Learning rate	3e-4
	Batch size	256
	Epochs	50
Metrics		
	True positive rate	62%
	Validation MCC	0.87
	Test MCC	0.72

Table 6.4: RNN hyperparameters used and best result achieved

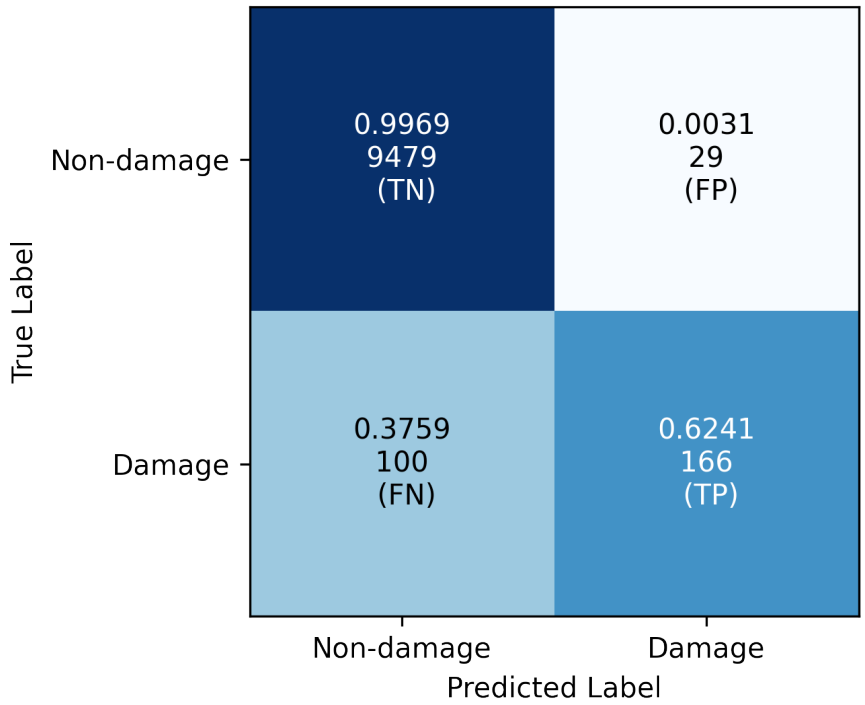


Figure 6.3: RNN confusion matrix test dataset result

6.4.3.2 LSTM

Hyperparameters		
	Learning rate	3e-4
	Batch size	64
	Epochs	500
Metrics		
	True positive rate	67%
	Validation MCC	0.85
	Test MCC	0.63

Table 6.5: LSTM hyperparameters used and best result achieved

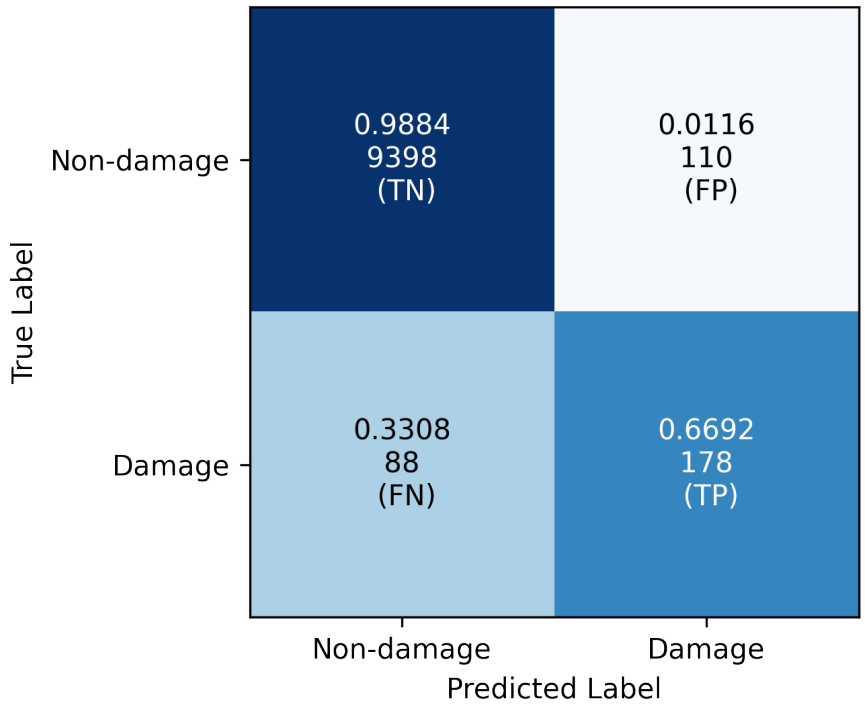


Figure 6.4: LSTM confusion matrix test dataset result

6.4.3.3 GRU

Hyperparameters		
	Learning rate	2e-4
	Batch size	256
	Epochs	300
Metrics		
	True positive rate	77%
	Validation MCC	0.83
	Test MCC	0.74

Table 6.6: GRU hyperparameters used and best result achieved

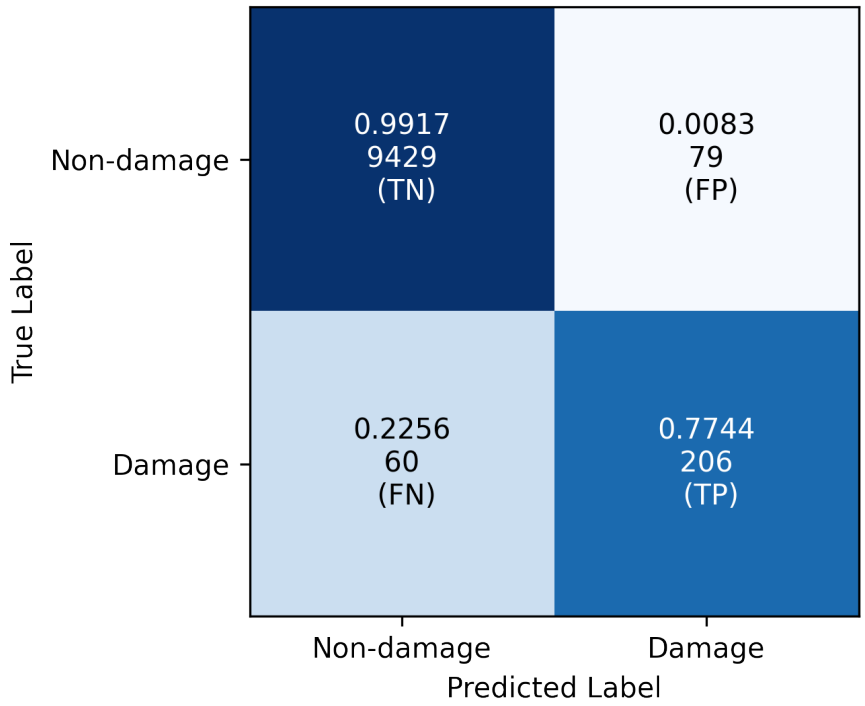


Figure 6.5: GRU confusion matrix test dataset result

6.5 Discussion of results

The objective stated in the Introduction chapter was not fulfilled. The goal would be to achieve 90% performance but the best result achieved was 82%.

The values are hardly distinguishable from each algorithm when discussing the final test MCC value. However, the LSTM algorithm result come as a surprise from the rest for being the only one below the 0.7 value. This disparity of 0.1 does come as surprise when compared to the MCC test value of the other RNN algorithms, with the simplified RNN algorithm version returning 0.72 and the GRU algorithm returning 0.74. Despite the results from the three approaches in the validation MCC value being similar to the LSTM algorithm, it could not generalize the information learned in the same way when faced with the test set.

The LSTM algorithm test MCC was the biggest remark but is followed by a number of others. The validation MCC value by the MLP algorithm is noteworthy, passing the 0.9 barrier. The feature engineering and selection gave the jump compared to the CNN and RNN algorithms input data. However the final result was equal to the rest of algorithms, despite learning the data better it was not capable of generalizing the test data better when compared to the rest.

As discussed in the introduction chapter, the final MCC value should not be the only metric to retain from this benchmark. The number of FP should also be a metric of analysis. In this component, two algorithms stand apart from the rest: RNN and CNN algorithms provide lower number of FP, with the CNN algorithm returning 32 FP and the RNN algorithm returning 29 FP. In this instance the RNN algorithm

wins by a low margin. However the case inverts when discussing the overall test MCC value.

In conclusion, feature engineering helped the MLP algorithm reach the maximum validation MCC value and true positive rate value but the test MCC was on par with the rest. The lowest test MCC value was returned by the LSTM algorithm despite being on par in terms of validation MCC value.

The best overall test MCC value belongs to the MLP algorithm. However, when discussing the FP values, CNN and RNN algorithms performed better despite reaching a lower MCC test value.

Chapter 7

Conclusion and Future Work

Of the objectives initially defined in the chapter Introduction one was successfully fulfilled while the other was not. The objective of going through the development cycle in the AI domain was fulfilled, the effort to accomplish this step involved defining two independent projects to be aligned according to the phases of the cycle to be executed. This objective provided insight and understanding on the development of an AI project.

The challenge of developing a neural network capable of detecting impacts with damage with a minimum true positive rate of 90% was not achieved. The best value recorded was 82% coming from an MLP algorithm and the lowest value recorded belongs to the RNN algorithm with 62%.

These figures were not surprising after the data analysis carried out in chapter Accelerometer sensor data understanding and analysis approach for damage detection allowed for the realization that the data was remarkably unbalanced, since the presence of data from events with damage is a small fraction of the total.

The presence of events in the test dataset never seen in the training dataset also contributed as a factor to a performance below expectations. Although the network was able to identify patterns in the training folder which could be verified with the validation results, the network was not capable of generalizing to new cases in the test folder. Despite these constraints the best result was not far from the initial goal.

For future work there is more to be explored in order to achieve the goal of 90% in true positive rate, namely the acquisition of more data from events with damage in order to increase the knowledge of the network so that it can recognize patterns more efficiently. Exploitation of the CNN 2D algorithm should also be a possibility in the case of scratch events, by being able to draw knowledge from the signal spectrogram as image recognition.

This objective was the most challenging from the two set initially not only in trying to achieve the best result but also to understand in how to plan and execute all the steps necessary to reach a valuable proposition.

Bibliography

Ali, A., & Eid, M. (2015). An automated system for accident detection. *2015 IEEE International Instrumentation and Measurement Technology Conference (I2MTC) Proceedings*, 1608–1612. https://doi.org/10.1109/I2MTC.2015.7151519

Al-Máadeed, N., Asim, M., Al-Máadeed, S., Bouridane, A., & Beghdadi, A. (2018). Automatic detection and classification of audio events for road surveillance applications. *Sensors (Basel, Switzerland)*, *18*. https://doi.org/10.3390/s18061858

Baumgärtel, H., Kneifel, A., Gontscharov, S., & Krieger, K. (2014). Investigations and comparison of noise signals to useful signals for the detection of dents in vehicle bodies by sound emission analysis. *Procedia Technology*, *15*, 716–725. https://doi.org/10.1016/j.protcy.2014.09.044

Cai, H., Hu, Z., Chen, Z., & Zhu, D. (2018). A Driving Fingerprint Map Method of Driving Characteristic Representation for Driver Identification. *IEEE Access*, *6*, 71012–71019. https://doi.org/10.1109/ACCESS.2018.2881722

Campo, I., Asua, E., Martínez, M. V., Mata-Carballeira, Ó., & Echanobe, J. (2018). Driving style recognition based on ride comfort using a hybrid machine learning algorithm*. *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, 3251–3258. https://doi.org/10.1109/ITSC.2018.8569722

Cheng, Z., Jeng, L.-W., & Li, K. (2018). Behavioral classification of drivers for driving efficiency related adas using artificial neural network. *2018 IEEE International Conference on Advanced Manufacturing (ICAM)*, 173–176. https://doi.org/10.1109/AMCON.2018.8614836

Cismas, A., Matei, I., Ciobanu, V., & Ca□u, G. (2017). Crash detection using imu sensors. *2017 21st International Conference on Control Systems and Computer Science (CSCS)*, 672–676. https://doi.org/10.1109/CSCS.2017.103

Collin, J., Davidson, P., Kirkko-Jaakkola, M., & Leppäkoski, H. (2019). Inertial Sensors and Their Applications [DOI: 10.1007/978-3-319-91734-4_2]. In S. S. Bhattacharyya, E. F. Deprettere, R. Leupers, & J. Takala (Eds.), *Handbook of Signal Processing Systems* (pp. 51–85). Springer International Publishing. Retrieved October 9, 2020, from http://link.springer.com/10.1007/978-3-319-91734-4_2

Domingos, P. (2012). A few useful things to know about machine learning. *Communications of the ACM*, *55*(10), 78–87. https://doi.org/10.1145/2347736.2347755

Domingos, P. (2017). *The Master Algorithm: How the Quest for the Ultimate Learning Machine Will Remake Our World* [OCLC: 981930098]. Penguin Books Ltd.

Faiz, A. B., Imteaj, A., & Chowdhury, M. (2015). Smart vehicle accident detection and alarming system using a smartphone. *2015 International Conference on Computer and Information Engineering (ICCIE)*, 66–69.

Feng, Y., Pickering, S., Chappell, E., Iravani, P., & Brace, C. (2018). Driving style analysis by classifying real-world data with support vector clustering. *2018 3rd IEEE International Conference on Intelligent Transportation Engineering (ICITE)*, 264–268. https://doi.org/10.1109/ICITE.2018.8492700

Foggia, P., Petkov, N., Saggese, A., Strisciuglio, N., & Vento, M. (2016). Audio Surveillance of Roads: A System for Detecting Anomalous Sounds. *IEEE Transactions on Intelligent Transportation Systems*, *17*(1), 279–288. https://doi.org/10.1109/TITS.2015.2470216

Foggia, P., Saggese, A., Strisciuglio, N., Vento, M., & Vigilante, V. (2019). Detecting Sounds of Interest in Roads with Deep Networks [DOI: 10.1007/978-3-030-30645-8_53]. In E. Ricci, S. Rota Bulò, C. Snoek, O. Lanz, S. Messelodi, & N. Sebe (Eds.), *Image Analysis and Processing – ICIAP 2019* (pp. 583–592). Springer International Publishing. Retrieved December 20, 2020, from http://link.springer.com/10.1007/978-3-030-30645-8_53

Gontscharov, S., Baumgärtel, H., Kneifel, A., & Krieger, K.-L. (2014). Algorithm Development for Minor Damage Identification in Vehicle Bodies Using Adaptive Sensor Data Processing. *Procedia Technology*, *15*, 586–594. https://doi.org/10.1016/j.protcy.2014.09.019

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning* [http://www.deeplearningbook.org]. MIT Press.

Hashimoto, W., Hirota, M., Araki, T., Yamamoto, Y., Egi, M., Hirate, M., Maura, M., & Ishikawa, H. (2019). Detection of car abnormal vibration using machine learning. *2019 IEEE International Symposium on Multimedia (ISM)*, 40–407. https://doi.org/10.1109/ISM46123.2019.00015

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, *9*(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Koch, M., & Bäck, T. (2018). Machine learning for predicting the impact point of a low speed vehicle crash. *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 1432–1437.

Li, Y., Li, X., Zhang, Y., Liu, M., & Wang, W. (2018). Anomalous Sound Detection Using Deep Audio Representation and a BLSTM Network for Audio Surveillance of Roads. *IEEE Access*, *6*, 58043–58055. https://doi.org/10.1109/ACCESS.2018.2872931

Ly, M. V., Martin, S., & Trivedi, M. (2013). Driver classification and driving style recognition using inertial sensors. *2013 IEEE Intelligent Vehicles Symposium (IV)*, 1040–1045. https://doi.org/10.1109/IVS.2013.6629603

Mantzekis, D., Savelonas, M., Karkanis, S., & Spyrou, E. (2019). Rnns for classification of driving behaviour. *2019 10th International Conference on Information, Intelligence, Systems and Applications (IISA)*, 1–2. https://doi.org/10.1109/IISA.2019.8900693

Marcillo, P., Barona López, L. I., Valdivieso Caraguay, Á. L., & Hernández-Álvarez, M. (2020). Modeling of a Vehicle Accident Prediction System Based on a Correlation of Heterogeneous Sources [DOI: 10.1007/978-3-030-50943-9_33]. In N. Stanton (Ed.), *Advances in Human Aspects of Transportation* (pp. 260–266). Springer International Publishing. Retrieved October 9, 2020, from http://link.springer.com/10.1007/978-3-030-50943-9_33

Matousek, M., Yassin, M., Al-Momani, A., Heijden, R. V. D., & Kargl, F. (2018). Robust detection of anomalous driving behavior. *2018 IEEE 87th Vehicular Technology Conference (VTC Spring)*, 1–5. https://doi.org/10.1109/VTCSpring.2018.8417777

McLoughlin, I., Zhang, H., Xie, Z., Song, Y., & Xiao, W. (2015). Robust Sound Event Classification Using Deep Neural Networks. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *23*(3), 540–552. https://doi.org/10.1109/TASLP.2015.2389618

Mitchell, T. M. (1997). *Machine learning* (1st ed.). McGraw-Hill, Inc.

Mnasri, Z., Rovetta, S., & Masulli, F. (2020). Audio surveillance of roads using deep learning and autoencoder-based sample weight initialization, 99–103. https://doi.org/10.1109/MELECON48756.2020.9140594

Moukafih, Y., Hafidi, H., & Ghogho, M. (2019). Aggressive driving detection using deep learning-based time series classification. *2019 IEEE International Symposium on INnovations in Intelligent SysTems and Applications (INISTA)*, 1–5. https://doi.org/10.1109/INISTA.2019.8778416

Mumcuoğlu, M. E., Alcan, G., Unel, M., Cicek, O., Mutluergil, M., Yılmaz, M., & Koprubasi, K. (2019). Driving behavior classification using long short term memory networks. *2019 AEIT International Conference of Electrical and Electronic Technologies for Automotive (AEIT AUTOMOTIVE)*, 1–6. https://doi.org/10.23919/EETA.2019.8804534

Naranjo, D., Roa, L., Reina-Tosina, L. J., & Estudillo, M. (2009). Optimization procedure for the impact detection thresholds in an accelerometer smart sensor. *2009 9th International Conference on Information Technology and Applications in Biomedicine*, 1–4. https://doi.org/10.1109/ITAB.2009.5394354

Nath, P., & Malepati, A. (2018). Imu based accident detection and intimation system. *2018 2nd International Conference on Electronics, Materials Engineering Nano-Technology (IEMENTech)*, 1–4. https://doi.org/10.1109/IEMENTECH.2018.8465309

Pai, A., Vernekar, V., Kudchadkar, G., Arsekar, S., Tanna, K., Rebello, R., & Desai, M. (2014). Real Time Collision Detection and Fleet Management System [DOI: 10.1007/978-3-319-03107-1_73]. In S. C. Satapathy, P. S. Avadhani, S. K. Udgata, & S. Lakshminarayana (Eds.), *ICT and Critical Infrastructure: Proceedings of the 48th Annual Convention of Computer Society of India- Vol I* (pp. 671–678). Springer International Publishing. Retrieved October 9, 2020, from http://link.springer.com/10.1007/978-3-319-03107-1_73

Parviainen, J., Colliri, J., Pihlstrom, T., Takala, J., Hanski, K., & Lumiaho, A. (2014). Automatic crash detection for motor cycles. *IECON 2014 - 40th Annual Conference of the IEEE Industrial Electronics Society*, 3409–3413. https://doi.org/10.1109/IECON.2014.7049003

Rosenblatt, F. (1957). *The perceptron, a perceiving and recognizing automaton project para*. Cornell Aeronautical Laboratory.

Saleh, K., Hossny, M., & Nahavandi, S. (2017). Driving behavior classification based on sensor data fusion using lstm recurrent neural networks. *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, 1–6. https://doi.org/10.1109/ITSC.2017.8317835

Sammarco, M., & Detyniecki, M. (2018). Crashzam: Sound-based Car Crash Detection: 27–35. https://doi.org/10.5220/0006629200270035

Sammarco, M., & Detyniecki, M. (2019). Car Accident Detection and Reconstruction Through Sound Analysis with Crashzam [DOI: 10.1007/978-3-030-26633-2_8]. In B. Donnellan, C. Klein, M. Helfert, & O. Gusikhin (Eds.), *Smart Cities, Green Technologies and Intelligent Transport Systems* (pp. 159–180). Springer International Publishing. Retrieved October 7, 2020, from http://link.springer.com/10.1007/978-3-030-26633-2_8

Savelonas, M., Mantzekis, D., Labiris, N., Tsakiri, A., Karkanis, S., & Spyrou, E. (2020). Hybrid time-series representation for the classification of driving behaviour. *2020 15th International Workshop on Semantic and Social Media Adaptation and Personalization (SMA*, 1–6. https://doi.org/10.1109/SMAP49528.2020.9248460

Selmanaj, D., Corno, M., & Savaresi, S. M. (2017). Hazard Detection for Motorcycles via Accelerometers: A Self-Organizing Map Approach. *IEEE Transactions on Cybernetics*, *47*(11), 3609–3620. https://doi.org/10.1109/TCYB.2016.2573321

Stefanakis, N., & Mouchtaris, A. (2015). A multi-sensor approach for real-time detection and classification of impact sounds. *2015 23rd European Signal Processing Conference (EUSIPCO)*, 2038–2042. https://doi.org/10.1109/EUSIPCO.2015.7362742

Supriya, M. S., & Gangadhar, N. D. (2020). Automotive Crash Detection Using Multi-sensor Data Fusion [DOI: 10.1007/978-981-15-3477-5_19]. In G. R. Kadambi, P. B. Kumar, & V. Palade (Eds.), *Emerging Trends in Photonics, Signal Processing and Communication Engineering* (pp. 149–155). Springer Singapore. Retrieved October 9, 2020, from http://link.springer.com/10.1007/978-981-15-3477-5_19

Surakul, K., & Smanchat, S. (2016). An accident detection technique using inertial measurement unit and odometry. *2016 13th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, 1–5. https://doi.org/10.1109/JCSSE.2016.7748899

Vaitkus, V., Lengvenis, P., & Zylius, G. (2014). Driving style classification using long-term accelerometer information. *2014 19th International Conference on Methods and Models in Automation and Robotics (MMAR)*, 641–644. https://doi.org/10.1109/MMAR.2014.6957429

Yee, T. H., & Lau, P. Y. (2018). Mobile vehicle crash detection system. *2018 International Workshop on Advanced Image Technology (IWAIT)*, 1–4. https://doi.org/10.1109/IWAIT.2018.8369671

Zeiler, M. D., & Fergus, R. (2014). Visualizing and Understanding Convolutional Networks [DOI: 10.1007/978-3-319-10590-1_53]. In D. Fleet, T. Pajdla, B. Schiele, & T. Tuytelaars (Eds.), *Computer Vision –*

ECCV 2014 (pp. 818–833). Springer International Publishing. Retrieved February 5, 2021, from http://link.springer.com/10.1007/978-3-319-10590-1_53