TÉCNICO LISBOA

Marine Acoustic Signature Recognition using Convolutional Neural Networks

Alexandre Martins Correia

Thesis to obtain the Master of Science Degree in

Mechanical Engineering

Supervisors: Prof. João Miguel da Costa Sousa
Dr. Guilherme Nuno Vasconcelos Beleza Vaz

Examination Committee

Chairperson: Prof. Duarte Pedro Mata de Oliveira Valério
Supervisor: Dr. Guilherme Nuno Vasconcelos Beleza Vaz
Member of the Committee: Inv. Bülent Düz

July 2021

Acknowledgements

I would like to give my thanks to Professor João Sousa for his help and guidance.

I am equally grateful to Dr. Engineer Guilherme Vaz and Engineer Miguel Vicente from the WavEC company for their unconditional whole collaboration and support in this work. In addition, I would like to thank to Msc. Érica Cruz for the insight provided in field of marine bioacoustic. Working with the WavEC team was a great opportunity, since this work was my first experience outside the academic context.

Thanks to all my friends from Instituto Superior Técnico.

And last but no least to my parents and my sister for their unconditional support.

Resumo

No mar existe uma grande diversidade de fontes sonoras: animais marinhos, fenómenos atmosféricos e atividades humanas. Em resposta às preocupações ecológicas e à necessidade de um maior controlo na navegação, o reconhecimento dessas fontes constitui um desafio. O objetivo deste trabalho consiste em construir um modelo que analise sinais acústicos captados por hidrofones e que os classifique de acordo com a fonte sonora. Assim, propõe-se a aplicação de uma rede neuronal convolucional (CNN) que recebe a representação do sinal em mel-espectrograma, dividido em pequenos intervalos (janelas), e o resultado do cálculo da primeira e segunda derivada do mesmo mel-espectrograma. Para cada janela, são atribuídos *class scores* pela CNN. Esta metodologia é aplicada a duas bases de dados formadas por sinais hidroacústicos. A primeira constituída por ruído de embarcações (base de dados ShipsEar), a partir da qual se pretende detetar a sua presença, diferenciando-as em função da dimensão. Usando o mel-espectrograma e o mel-espectrograma em conjunto com a primeira e segunda derivada, obtém-se, respetivamente, uma exatidão de 83.2% e 88.8%. A segunda base de dados complementa a primeira com vocalizações de golfinhos e de baleias. Três técnicas de aumento de dados (*time stretching, pitch shifting* e *time shifting*) são estudadas. Aplicadas individualmente, ou simultaneamente, permitem um melhor desempenho em relação ao modelo sem aumento de dados. A influência da dimensão de janelas é também estudada, através da construção de cinco modelos em que a dimensão da janela varia entre 0.22 s e 1.97 s. A percentagem de exatidão situa-se entre os 66.2% e os 78.3%.

Palavras-chave: Reconhecimento de sinais hidroacústicos, rede neuronal convolucional, mel-espectrograma, base de dados ShipsEar, ruído de embarcações, vocalização de animais marinhos

Abstract

In a marine environment, there is a great diversity of sound sources: marine animals, natural phenomena and man-made activity. Differentiating these sources is an important response to ecological challenges and to ensure better control of the coastline. The current work aims to devise a model which analyses acoustic signals from hydrophones and classifies them according to the sound source. To achieve this, a convolution neural network (CNN) is proposed as a classifier, using the mel spectrogram representation divided into small intervals (windows) of the acoustic signal and its derivatives as input. Class scores are assigned to each window by the CNN. The developed methodology is applied to two different datasets composed of hydroacoustic data. The first comprises vessel noise data (ShipsEar dataset), where the objective is to detect the presence of vessels and distinguish the vessels according to the size. A classification accuracy of 83.2% and 88.8% is achieved using the mel spectrogram and the mel spectrogram plus its first and second derivatives as features, respectively. The second dataset complements the previous one by adding dolphin and whale vocalizations. Three data augmentation techniques (time stretching, pitch shifting and time shifting) are studied. Whichever of the three techniques is used individually outperform the model without data augmentation. This is equal true when all three techniques are used simultaneously. The impact of the window length is also studied, with five different models being creating where the window length varies between 0.22 s and 1.97 s. The classification accuracy ranges between 66.2% and 78.3%.

Keywords: Hydroacoustic signal recognition, convolutional neural network, mel spectrogram, ShipsEar dataset, vessel noise, marine animal vocalizations

Contents

List of Figures

List of Tables

List of Acronyms

ADC Analog-to-digital converter

AR Autoregressive

ASR Acoustic Signal Recognition

CNN Convolutional Neural Network

ConvNet Convolutional Neural Network

DNN Deep neural network

DSP Digital Signal Processing

FFT Fast Fourier Transform

GD Gradient Descent

KDD Knowledge discovery in databases

MFCC Mel Frequency Cepstral Coefficient

ML Machine Learning

PSD Power Spectral Density

ReLU Rectified Linear Unit

SOFAR Sound Fixing and Ranging

STFT Short Time Fourier Transform

List of Symbols

$\#frames$ Number of frames in the STFT matrix

$\hat{y}$ Predicted output

α Learning rate

c Mel spectrogram coefficients

D Depth

F Filter size

H Height

k Frequency bin

K Number of filters

L Loss function

M Window function length of STFT

N Input signal length

O Overlap length between segments

p Number of epochs

P Zero-padding

R Hop size between segments

S Stride

sr Sampling rate

w Window function of STFT

W Width

X STFT of a signal

x Signal in time domain

y Desired output

Programs

Python version 3.7.1
Audacity (digital audio editor)

Chapter 1

Introduction

1.1 Objectives and motivation

The contact with the sea is often perceived as a release from daily stress. However, this apparent calm hides a scenario which is far from quiet. In fact, sea noise is already a matter of concern, because apart from natural sounds (*e.g.* rain, waves, earthquakes) and animal vocalizations, sounds produced by human activities, such as shipping and energy exploration, have been rising over recent decades. Man-made activities have contributed to the increase in the noise pollution, which has a negative impact on the aquatic ecosystem.

The amount of sounds being produced has increased the interest in marine acoustic signature recognition in the last few years, which can be seen as an important tool in several scientific areas. For marine researchers, this is useful in the characterization of marine life, such as identifying fish migration or studying how sea traffic affects the behaviour of marine animals. Moreover, another important application is the coastal control of sea traffic. Sometimes there are vessels or submarines that do not share their locations, threatening maritime surveillance. Thus, marine acoustic signature recognition complements other ocean monitoring techniques based on underwater image classification [1][2][3], and can even complement maritime surface imaging from sources, such as from cameras in ports or drones [4][5]. We should note that underwater imaging is limited to less than fifty meters at best, whereas sound waves can travel up to as much as thousands of kilometers in seawater.

Therefore, the development of a model that can identify natural sounds, animal vocalization or man-made activity efficiently, based on underwater acoustic recordings, is extremely useful in countries with an extensive coastline. In the same way that there are microphones for collecting sounds from the air, there is a specific underwater device to detect ocean sounds: the hydrophone, which is the most suitable device for the operation of this task. This device converts underwater sounds into electrical signals and it can be installed at different locations. The data collected by the hydrophones is converted to a mel spectrogram representation and classified using a convolution neural network (CNN). The model assigns scores to the default classes over time. Figure 1.1 shows a simplified scheme of the process developed to recognize

marine acoustic signatures.

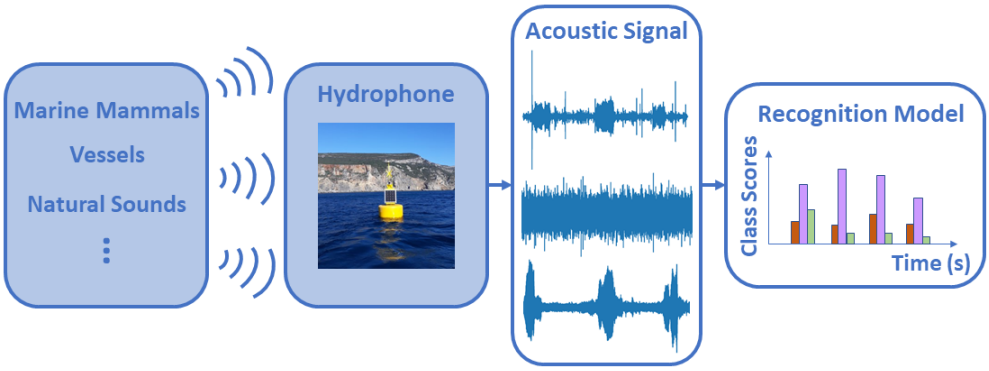

Figure 1.1: Marine acoustic signature recognition

1.2 Related work

Over time, researchers have conducted various models for detection and classification of marine acoustic signatures. These operations started to be carried out by experienced sonar operators that are experts in distinguishing types of vessels based on acoustic analysis, especially when the operation is associated with maritime safety. Sonar technicians are highly qualified and with an extensive experience, yet human error can always happen. In addition, it is a low automated activity, where the operator has to totally focus on a single activity. This is why recognition techniques of vessel sounds and animals vocalizations have been developed. These are usually split into two phases. Firstly, the acoustic signals undergo to processing techniques in order to extract features of interest, the so-called feature space, and, then, classification algorithms (classifiers) are applied using as input the features extracted. In general, the processing techniques consist in converting the signal to the frequency domain.

Some approaches are based on the assumptions that a particular known model can fit the data correctly. Since the model coefficients may represent the power spectral density (PSD), these can be seen as useful features for classification. One of the models widely used is the autoregressive (AR) model. Huang et al. [6] proposed a method to classify noise from four different ships using the AR model. However, AR model poles are used as features instead of the coefficients, with the justification that coefficients will vary greatly with the change of environment and position of ships. The nearest neighbour algorithm is used as classifier. Bennett et al. [7] has also used the AR model, but this time using its coefficients as feature space. In this study, another processing method was applied, based on the wavelet transform, considered a better option in relation to the short-time fourier transform (STFT). STFT gives an analysis of the signal in the time and frequency domain concurrently, which results in a spectrogram. The wavelet transform contains similar information, but unlike STFT, it allows the analysis of the signal with different resolutions in frequency and

time domains. The AR coefficients and the average energy contained in the wavelet coefficients were used as inputs to a neural network, in order to classify geological processes (earthquake data) and biological signals of five different species of whales.

Another common feature extraction approach is based on computing the PSD with the Fast Fourier Transform (FFT) [8], [9], or using Welch's method [10]. In fact, the PSD is quite a useful tool to extract the characteristics of a signal, because it may tell at which frequency ranges variations are strongest. However, PSD does not give information about the frequencies at each instant of time, since it computes a statistical average of the whole signal. Therefore, it is only recommended for stationary signals, whose variance and mean do not change with time. In the cases here dealt with, underwater signals are always non-stationary. As an alternative, the feature space can be generated by the STFT resulting in a spectrogram, which is a 2D image containing the noise signal frequency content changing with time. In this way, the implementation of the spectrogram as the feature space permits the application of a new classification algorithm in the field of acoustic signal recognition (ASR), known as CNNs. These are typically used for image recognition [11], and in here, CNNs try to find patterns in the spectrogram which are common to the same class.

In the work of Garcia et al. [12], in which five distinct classifiers were studied in order to distinguish fin whale vocalizations, the application of CNNs using spectrograms as inputs is highlighted. The same methodology was used by Harvey et al. [13] in the detection of humpback whales and by Belghith et al. [14] in the classification of underwater acoustic signals from widely diverse sources (natural sounds, animal vocalization and man-made activity), which is in some respect similar to the objective of this work.

Apart from hydroacoustic signal recognition, a considerable number of studies have been dedicated to ASR of a wide variety of sounds. UrbanSound8k [15] is a known dataset in this area, which consists of a fairly complete data of urban sounds. Several studies have been performed using this dataset, where spectrogram and CNNs have an important role [16][17]. These two studies provide the basis for the methodology that was developed to be applied to marine environment acoustic noise.

1.3 Thesis outline

This thesis is organized as follows:

Chapter 2 | Machine learning and knowledge discovery process: The general concepts of machine learning (ML) are described and it is explained how ML is incorporated in the knowledge discovery process. In addition, a theoretical view of the main components of neural networks and, in more detail, CNNs are presented.

Chapter 3 | Sound waves and frequency analysis: Theoretical background about sound waves is described, where the main characteristics of sound wave propagation in the ocean are mentioned. Additionally it is shown how the sound waves are converted to a digital signal. Moreover, some techniques for analysing the signal in the frequency domain are presented.

Chapter 4 | Knowledge discovery process of recognition framework: The recognition framework is introduced, explaining in detail the main steps in the feature extraction and modeling process. The evaluation metrics of the classifier performance are also described. In addition, a testing of the recognition framework with a known dataset of non-marine sounds is analysed.

Chapter 5 | Application of the recognition framework in the marine environment: The two datasets composed of hydroacoustic data used are described and several models applying the recognition framework are assembled for these two datasets. The results obtained are presented.

Chapter 6 | Conclusions: An overall inference of the work performed is described and there are recommendations for future work.

Chapter 2

Machine learning and knowledge discovery process

Machine learning (ML) is an application of artificial intelligence that makes it possible for machines to automatically learn from experience, without being programmed explicitly. Problems that are hard to define by rules are too complex for traditional programming techniques, but not for ML algorithms, since they are capable of finding relationships within data and make smart predictions based on that. Therefore, in ML, building up a predictive model always requires a data set which is divided into two groups: training data and testing data. The former corresponds to the majority of the dataset and it may or may not be associated with the desired labels, depending on the type of learning (as explained in the next paragraph). So, training examples are used to create a predictive model and optimize its parameters based on finding relationships between patterns. Once the model is generated, it is applied to testing data, in order to predict the unknown labels.

According to the type of learning, ML systems can be classified into three categories: supervised learning, unsupervised learning and reinforcement learning [18]. Supervised and unsupervised learning are the two biggest types of ML and the main difference between them consists in the prior knowledge of the labels in the training set. Thus, in supervised learning, models learn from a labelled training set, unlike in unsupervised learning, that models discover patterns from unlabelled data. Reinforcement learning is based on a feedback system that uses rewards and punishment signals according to positive and negative behaviours.

ML can be incorporated in a high-level process known as knowledge discovery in databases (KDD). Fayyad et al. [19] defined KDD as "the overall process of discovery knowledge from data". The KDD is directly related to data mining, which is a step in the knowledge discovery process. Data mining applies ML, pattern recognition and statistics techniques of classification, clustering, regression, etc., in order to discover patterns that would not be immediately apparent in the data. In Figure 2.1, the KDD process is illustrated according to [19]. The five steps are defined as follows:

1. **Data selection**: divided into two phases. Firstly, developing and understanding the application domain and identifying the goal of the problem. Secondly, creating a data set that matches with the objective defined (target data).

2. **Preprocessing**: converting the "raw data" into a set of features.

3. **Transformation or Feature selection**: selecting the features that are more relevant for the data outcomes.

4. **Data mining or Modeling**: divided into three steps. Firstly, selecting the modeling task that best performs the goals defined in the first step. For example, classification, clustering, regression and so on. Secondly, choosing a proper model to extract knowledge from the reduced data. Finally, applying the selected model to generate the patterns in a particular representation form.

5. **Interpretation/Evaluation**: evaluating the outcomes of the model and making any changes in one or more of the previous to improve the process.

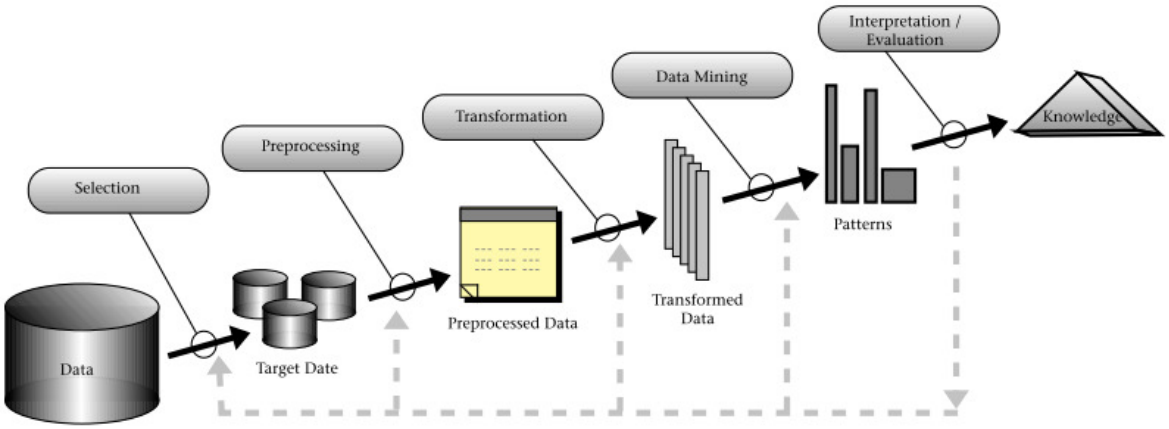

Figure 2.1: KDD process. Taken from [19]

2.1 Neural networks

One of the set of algorithms used in ML is neural networks. The structure of neural networks is inspired in the human brain and the objective is to capture non-linear patterns in training data, in order to predict the outputs for the testing data.

The algorithms, in neural networks, are made up of layers of neurons which can be classified in three different categories: input layers, hidden layers and output layers. The input layer receives the input (a vector or a matrix) and the output layer predicts the final output. Between the input and output layers, there are the hidden layers. Each neuron of one layer is connected to the neurons of the next layer.

In Figure 2.2, the architecture of a Multi-layer Perceptron, a standard neural network model, is represented. When there is more than one hidden layer, it is called Deep Neural Network (DNN). As may be seen, each connection between neurons is assigned a numerical value known as weight (w). Each weight is multiplied by the value associated to each neuron and the obtained value is sent as input to the next

hidden layer. Each of these neurons is associated a numerical value, called bias (δ), which is added to the input sum.

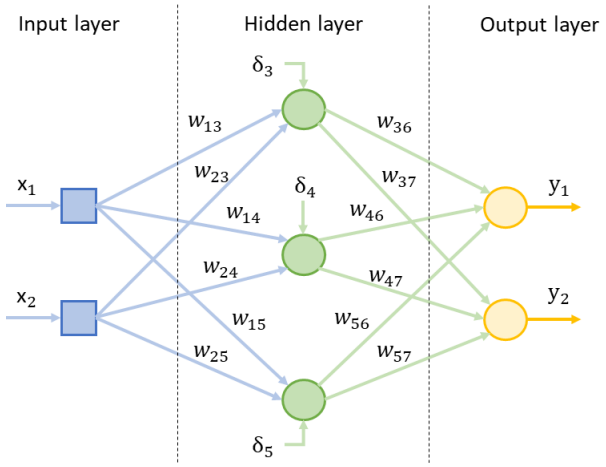

Figure 2.2: Multi-layer Perceptron

For instance, the output of the first neuron in the hidden layer of Figure 2.2 is computed as follows:

$$a_3 = \begin{bmatrix} w_{13} & w_{23} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \delta_3. \tag{1}$$

Then, this value is passed through the activation function, which is attached to each neuron and it determines if the neuron will get activated or not. When a neuron is activated, it transmits data to the neuron of the next layer over the channels, so data is propagated through the network. This is called **Forward Propagation**. In the output layer, the neuron with highest value corresponds to the prediction of the network.

In the training process, the network also has information about the actual output for each input. So, the predicted output is compared to actual output to compute the error in the prediction. The computed error is important because the information is transferred backward through the network and, hence, the weights can be adjusted in order to minimise the error. This is known as **Back Propagation** [20].

This cycle of Forward Propagation and Back Propagation is iteratively performed with multiple inputs. The training process ends when the assigned weights can best predict the actual training data. The parameters (weights and bias) are the numerical values that are adjusted through training the network.

2.2 Convolutional neural networks

There are various kinds of neural network architecture that are differentiated in the way that neurons are connected. Convolutional neural network (CNN or ConvNet) is a particular type of Neural Network that are typically used for image recognition, but can also be applied to other fields, such as for audio recognition. The CNN is characterized for requiring a low level of data pre-processing compared to other neural networks.

The advantage of CNNs in comparison to other types of Neural Networks is based on the way they cope with the input. A digital representation of colour images consists of three channels, *i.e*, three matrices. Thus, the first hidden layer of an ordinary Neural Network, which is fully connected to all neurons of the input layer would require a vast number of parameters, rapidly leading to overfitting. In CNNs, the input data goes through convolutional layers instead of hidden layers, in order to reduce the amount of parameters required. Basically, ConvNet arquitecture reduces the full image into a single vector of class scores.

Unlike hidden layers, convolutional layers have neurons arranged in 3 dimensions: **Width** (W), **Height** (H) and **Depth** (D). So, the number of neurons in each layer is given by **WxHxD**. Each neuron connects to only one local region of the input, which is known as **Local Connectivity**. This connectivity is applied by a receptive field (filter or kernel). Moreover, in convolutional layers, the neurons of each depth slice have to share the same set of weights, in order to reduce the total number of parameters. This is called **Parameter Sharing** [21].

There are four hyperparameters (parameters used to control the learning process) related to the receptive field that controls the number of neurons in the convolutional layer, *i.e*, the spatial size of the convolutional layer output. The first hyperparameter is **number of filters** (K) which sets the depth of the output volume. The second one is the **filter size** (F). The third hyperparameter is the **stride** (S) that indicates how the filter is slid along the input. For instance, if S=1, the filter moves one pixel at a time, if S=2, the filter moves two pixels and so on. Finally, the fourth hyperparameter is the amount of **zero-padding** (P), since it may be desirable to pad the input volume with zeros around the border. Thus, the spatial size of the output volume could be represented as a function of these four parameters. Assuming W as the input volume size, the output size is given by:

$$Output = \frac{W - F + 2P}{S} + 1. \tag{2}$$

In the next figure, an example of a convolutional layer connected to an input layer is shown in order to understand in more detail the characteristics of convolutional layers.

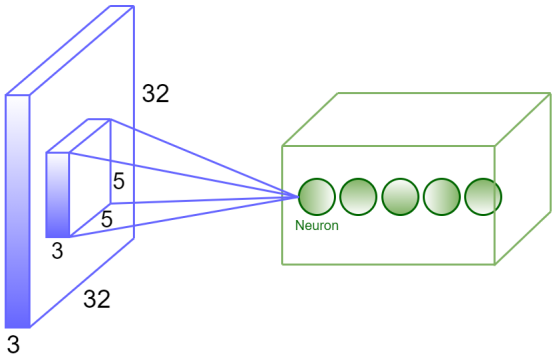

Figure 2.3: Convolutional Layer

As can be seen, the input size is equal to 32 x 32 x 3 ($W_1 = 32, H_1 = 32, D_1 = 3$) and the convolutional layer has 5 depth slices ($D_2 = K = 5$). The filter size is 5x5 (F=5) and let's assume that a stride value equals to 1 ($S = 1$) and no zero-padding (P=0). The number of weights per filter is given by $F \cdot F \cdot D_1$ and it leads to a total of $(F \cdot F \cdot D_1) \cdot K$ weights and K biases. So, there are a total number of $(5 \cdot 5 \cdot 3) \cdot 5 + 5 = 380$ parameters. Regarding the output volume, it may by obtained by applying Equation 2:

$$W_2 = (W_1 - F + 2P)/S + 1 = (32 - 5 + 0)/1 + 1 = 28. \tag{3}$$

$$H_2 = (H_1 - F + 2P)/S + 1 = (32 - 5 + 0)/1 + 1 = 28. \tag{4}$$

Therefore, the output volume is 28 x 28 x 5 and hence, there are 3920 neurons.

In ConvNets, it is frequent to insert an additional layer between two convolutional layers an additional layer, called the max-pooling layer. This is to control the overfitting, because it progressively reduces the spatial size of the representation and, hence, the amount of parameters. Thus, pooling layers resize the input using a filter which takes the maximum over a single region. In Figure 2.4, there is a practical example of a max-pooling layer with a filter size of 2 x 2 and stride 2. As a result, an input 4x4 is reduced to a matrix 2 x 2.

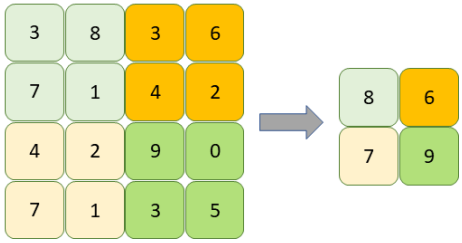

Figure 2.4: Max pooling with 2 x 2 filters and stride 2

Therefore, ConvNet architectures are made up of convolutional and max-pooling layers. In the final part, there are always a few fully-connected layers, as in simple neural networks, where the last layer holds the final output.

2.3 Activation and loss functions

In the first two sections of this chapter, the key components of neural networks and CNNs were introduced. In this section, two important components are studied: the activation function and the loss function.

CNN math operations associated to neurons are the same as for other neural networks. In each neuron, there is a dot product of the weights associated to the neuron with the input, followed by a sum of the bias. Then, the result passes through an activation function. Figure 2.5 shows this operation for one neuron.

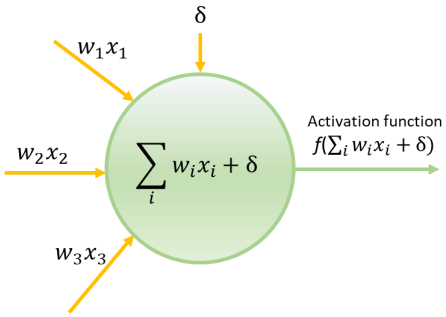

Figure 2.5: Mathematical operation of a neuron

The **activation functions** are very important to the neural network, since they provide the ability to learn patterns with non-linear properties, and this corresponds to most physical phenomena. In CNNs, the activation functions are applied in the neurons of the convolutional layers and fully-connected layers. There are a vast number of activation functions. The Rectified Linear Unit (ReLU) function [18] is usually the preferential choice. The ReLU is a common simple non-linear activation function and it is defined as $f(x) = max(0, x)$. However, the output layer usually uses a different activation function depending on the purpose of the model. In this work, the objective is to recognize a variety of acoustic signals using CNNs. So, it corresponds to a multi-class classification problem. The softmax function is usually the best option for these cases and it was the score function applied to the last (fully-connected) layer of the developed models. The softmax function is given by:

$$f_j(z) = \frac{e^{z_j}}{\sum_k e^{z_k}}. \tag{5}$$

Following the softmax expression, the raw class scores that come from the last fully-connected layer (z vector) are compressed into a vector of values between zero and one that sum to one. The values from the softmax classifier will be seen as probabilities. However, these probabilities could be more peaky or diffuse using multiples of the same weights. If the softmax function is applied using, for example, a three classes model, with the output of the last layer being $\begin{bmatrix} 1 & 2 & 4 \end{bmatrix}$, the result is given by:

$$\left[\frac{e^1}{e^1+e^2+e^4} \quad \frac{e^2}{e^1+e^2+e^4} \quad \frac{e^4}{e^1+e^2+e^4}\right] = \left[0.042 \quad 0.114 \quad 0.844\right]. \tag{6}$$

Now, considering that the weights are divided by 2 and the bias is equal to zero, the output of the last layer becomes smaller by half: $\left[0.5 \quad 1 \quad 2\right]$. When applying the softmax function:

$$\left[\frac{e^{0.5}}{e^{0.5}+e^1+e^2} \quad \frac{e^1}{e^{0.5}+e^1+e^2} \quad \frac{e^2}{e^{0.5}+e^1+e^2}\right] = \left[0.140 \quad 0.231 \quad 0.629\right]. \tag{7}$$

The class scores are not the same as in the first example, though the differences between the outputs in the last layer still have the same proportions. Therefore, the probabilities should be considered as confidence values [22].

As an illustrative example, in Figure 2.6, the softmax function is applied into a three classes classification model. According to scores given by the softmax function, class C is the one chosen by the model, since it corresponds to the highest probability.

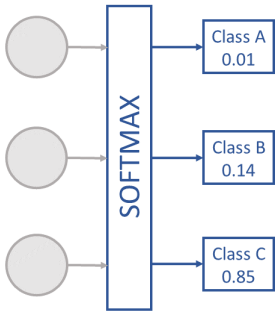

Figure 2.6: Output layer of a multi-class classification

The next step is to apply the **loss function** (L) in order to quantify how well the class scores match the desired output. Again, the loss function used depends on the objective of the model. For example, in a regression task, the mean square error is usually the most suitable. In the case of multi-class classification, cross-entropy is often a better option. Cross-entropy loss can be computed by the following formula:

$$L_{cross-entropy}(\hat{y}, y) = -\sum_i y_i ln(\hat{y}_i), \tag{8}$$

where $\hat{y}$ are the predicted outputs and y the desired outputs. The objective is to minimize L.

In order to clarify the cross-entropy loss calculation in Figure 2.6 the class scores after applying the softmax function are $\hat{y} = \left[0.01 \quad 0.14 \quad 0.85\right]$. Assuming the desired outputs are $y = \left[0 \quad 1 \quad 0\right]$. The cross entropy loss would be:

$$L_{cross-entropy}(\hat{y}, y) = -0 \times ln(0.01) - 1 \times ln(0.14) - 0 \times ln(0.85) = 1.97. \tag{9}$$

The cross-entropy loss is too high, yet this was expected since the highest number of the predicted output does not correspond to the desired output. This means that the cross-entropy loss has to be brought down, which leads us to the next step: Optimization.

2.4 Optimization

The loss function quantifies how close the predicted output is from the desired one to a particular set of weights. The goal of optimization is to find out the best set of weights in order to minimize the loss function. There are several ways to perform optimization. The gradient descent (GD) is used, which is the most common optimization algorithm. Basically, the GD algorithm iteratively computes the gradient of the lost function and performs a parameter (weights and bias) update taking into account the gradient, and using a learning rate (α) set by the model developer. The update of parameters follows the next formula:

$$\omega_{jk}(p+1) = \omega_{jk}(p) - \alpha \nabla L(\omega_{jk}), \tag{10}$$

where p is the number of epochs, which defines the number of times that the learning algorithm passes through the entire training set.

In this work, mini-batch GD [23] is used, which consists in dividing the training set into batches, and, instead of using the entire training set to update the parameters, this is performed for each batch. Every batch consists of a number of samples (previously defined) randomly selected from the training set without repetition. The training process finishes when all batches of all epochs are completed.

Chapter 3

Sound waves and frequency analysis

3.1 Characteristics of sound waves

When someone speaks or plays an instrument, sound waves are produced. These waves are a form of energy which needs a medium to travel, such as a gas, liquid or solid. A wave is a vibratory disturbance of molecules in a medium that transfers energy between two points without matter being transferred. Waves can be classified as transverse or longitudinal depending on the movement of the particles in a medium.

In a transverse wave, the particles vibrate perpendicularly to the direction of propagation of the wave. These waves only propagate in solid and liquid mediums. For instance, earthquakes are a transverse wave. On the other hand, in a longitudinal wave, the particles move parallel to the direction of propagation of the wave. These waves are produced in a solid, liquid and gaseous medium. Sound is a good example of this type of waves. Longitudinal waves are characterized for producing successive compressions and rarefactions in the medium. Compression is when the particles are close together due to the rise of the pressure. Rarefaction is when the particles are more spread out.

3.2 Sound waves in the ocean

The behaviour of sound waves depends on the characteristics of the medium. In water, the propagation of sound is much faster and there are less energy losses than in air. This is due to the density of water being much greater than the density of air, thus making for higher concentration of particles per volume. Besides the density of the medium, the distance travelled by the sound waves also depends on its temperature and pressure.

According to how temperature varies with the depth, the ocean can be divided into different regions. The top surface layer is called the epipelagic zone which is, in general, the warmer region. Then, there is the thermocline layer, which is the transition from the surface layer to the deep ocean. In this layer, the temperature drops rapidly and it occurs at different depths around the world. Finally, the temperature reaches its minimum at the bottom of the thermocline layer and it remains stable in the whole deep ocean region [24]. Figure 3.1 represents the temperature profile within the mid-latitude regions.

Therefore, when sound waves are produced in ocean, they undergo speed variations according to depth. As it may be seen in Figure 3.1, along the thermocline layer the temperature drops, so the speed decreases, which makes the sound waves to be refracted downwards [25]. At the bottom of the thermocline layer, sound reaches its minimum speed. From that point on, through the deep ocean region, the pressure turns into the dominant factor, since the temperature becomes stable. Once the pressure keeps increasing, the sound speed increases too, causing the sound waves to refract upwards. This upward and downward refraction creates a sound channel, known as Sound Fixing and Ranging (SOFAR) channel [26], which allows sound to propagate several miles without significant losses of energy and yet still be easily perceived by a detection device.

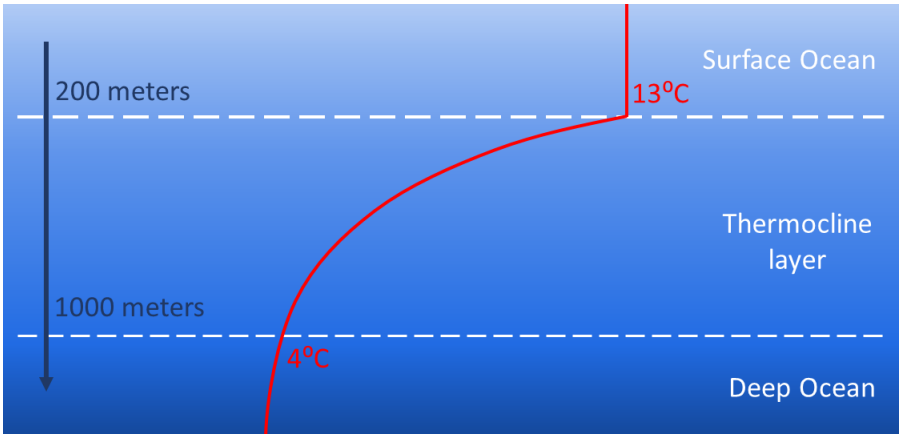

Figure 3.1: Ocean temperature profile within the mid-latitude regions
Based on image from NOAA

In addition, we should take into account that the propagation of sound waves differs depending on the depth of the sea bed. In shallow places (water depth until approximately 100 m), there are a lot interactions between the surface and the bottom of the ocean and, hence, there are higher sound waves energy losses. This is in contrast with deeper locations (1000 m), where these interactions are lower, causing less energy losses and making the sound waves travel longer distances.

3.3 Sound waves detection

The ocean is not silent, there is almost a constant noise due to many marine animals, man-made activity, and meteorological phenomena, such as waves, currents or rain. There is specific underwater device to detect ocean sounds: the hydrophone. This device is a passive sonar which differs from an active sonar since it is only used to detect sound waves, without emitting any signals [27].

The hydrophones are extremely important for this work, since it permits collection of the required data to develop the recognition model. The characteristics of the ambient noise recorded depends on its geographic

location. For example, if the hydrophone is located in the Baltic sea, where there is maritime traffic with several types of vessels, the noise sea level would be higher than in the Atlantic ocean, where there is still an intense maritime traffic but the sound sources are not so close.

Since the sound waves are recorded by the hydrophone as analog signals, these have to be converted to digital signals, which makes it easier to process and store. Looking at Figure 3.2, firstly, the hydrophones produce electrical current when subjected to changes due to underwater pressure, thus, converting the acoustic waves into analog electrical signals. Secondly, an analog-to-digital converter (ADC) is used. Once in digital format, digital signal processing (DSP) methods can be applied.

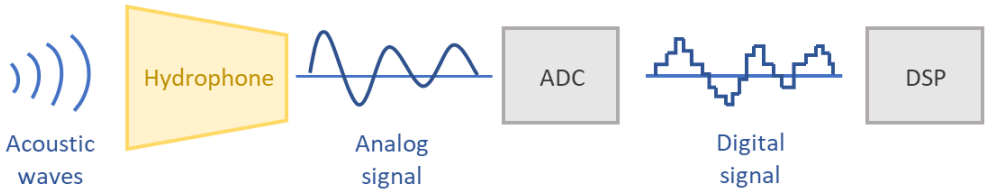

Figure 3.2: Conversion of acoustic waves into a digital signals

3.4 Spectrogram

Frequency or spectral analysis is one of the DSP domains, which consists of analysing the frequency components contained in a time-domain signal. To do this, a power spectral density (PSD) (or power spectrum) estimation has to be performed, in order to extract the frequency content of the signal. There are several techniques to estimate the PSD. These can be divided into two groups: the parametric and non-parametric methods. The former are based on using the model to estimate the PSD and it is also known as the model-based approach. This type of methods require prior knowledge about the data or the model, as it is the case of the Autoregressive model, the Moving Average model or the Autoregressive Moving Average model. Once the model is known, one can estimate its parameters from the observed data and thus compute the power spectrum [28]. On the other hand, nonparametric methods make no assumption about how the data is generated. This type uses the Fourier Transform as the Fast Fourier Transform (FFT) to obtain the PSD estimation. The FFT is an efficient algorithm for computing the Discrete Fourier Transform [29].

However, analysing a signal solely in the frequency domain will have some drawbacks, since PSD computes a statistical average of the whole signal, neglecting the specific frequency in each particular instant [30]. This leads us to another DSP domain: Time-frequency analysis. In this analysis, the spectrogram is used, which allows an analysis of the frequency content over time. It is a two-dimensional graph that represents the PSD for each instant of time for a given acoustic signal. The x-axis refers to time and the y-axis refers to frequency. Moreover, the amplitude of each frequency in each instant of time is usually represented

with colours in the spectrogram.

The most common method to compute the spectrogram is the short-time fourier transform (STFT), which consists in sliding the signal over a window function of length M, in order to divide it into shorter segments (frames) which usually overlap, and computing the discrete Fourier Transform of the windowed data [31]. The number of columns in the STFT matrix corresponds to the number of frames and it is given by:

$$\#frames = \frac{N - O}{M - O} \tag{11}$$

where N is the input signal length and O is the overlap length between segments.

In each time frame m, the STFT expression can be written as:

$$X_m(k) = \sum_{n=-\infty}^{\infty} x(n)w(n - mR)e^{-j\frac{2\pi kn}{N}} \tag{12}$$

where:

- x(n) is the input signal at sample n;
- w(n) is the window function with length M (*e.g*, Hann, Hamming, etc.);
- R is hop size between segments ($R = M - O$);
- k is the frequency bin (0, sr/M, $(2 * sr)/M$, ..., $sr/2$);
- sr is the sampling rate.

The STFT returns a matrix, which specifies the complex amplitude of the signal associated to each frequency in each time frame. Then, the spectrogram is computed as the magnitude squared of the STFT elements, as shown in the following expression:

$$Spectrogram(x(n)) = |X_m(k)|^2 \tag{13}$$

The spectrogram is usually converted to decibels.

In Figure 3.3, the signal waveform from a humpback whale vocalization and the respective spectrogram are presented. The higher amplitudes of the signal correspond to the intervals of the humpback whale vocalizations, which are also visible in the spectrogram.

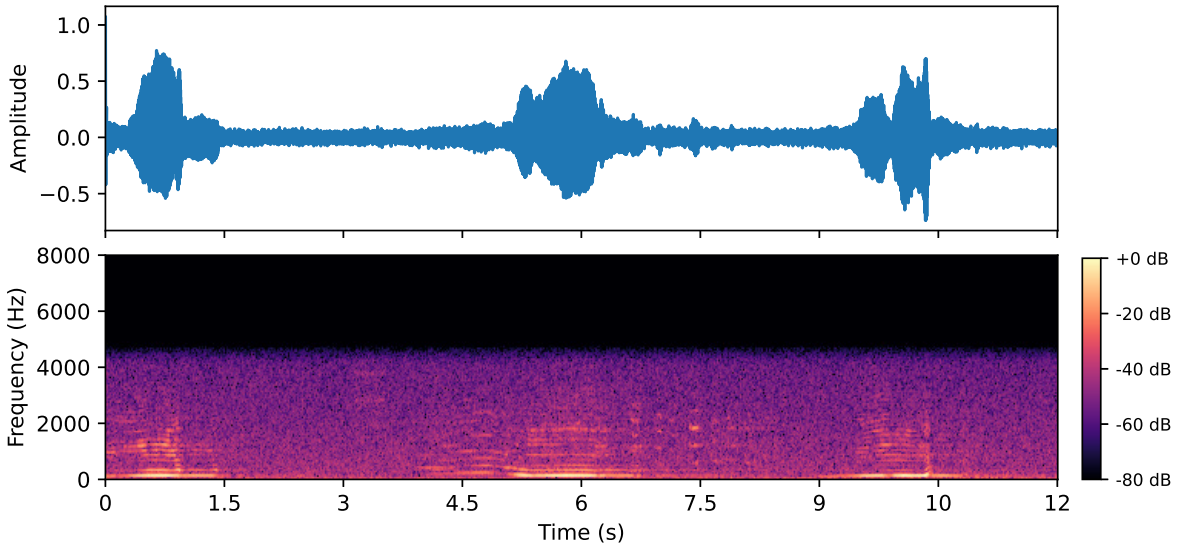

Figure 3.3: Signal waveform and spectrogram of a humpback whale vocalization

3.5 Mel Spectrogram

In spectrograms, frequency is expressed linearly. Thus, a jump from a lower to a higher frequency in low frequencies has the same relevance as the same jump in high frequencies. However, humans perceive frequency logarithmically, *i.e* we have a better resolution at lower frequencies than at higher frequencies. For instance, for humans, two musical notes from the high frequencies region separated by a given interval are closer in pitch than two notes from the lower frequencies with the same frequency interval. A filter bank, such as mel filter bank, can be applied in order to approximate the spectrogram to human auditory perception.

The conversion of frequency to mel scale consists of three steps. Firstly, the number of filter banks has to be defined. It is a parameter that varies greatly, but 40, 60, 90 or 128 bands are the most typical values. Secondly, the mel filter banks are computed as triangular filters [32]. Finally, they are applied to the spectrogram. In other words, the mel spectrogram is computed by the matrix multiplication between the mel filter banks and the spectrogram.

In Figure 3.4, the spectrogram is of the same humpback whale vocalization as in Figure 3.3 , but now the frequencies have been converted into a mel scale with 60 bands. In contrast to the linear scale, as seen in Figure 3.3, in mel scale lower frequencies are shown in more detail than higher frequencies. Therefore, the mel scale corresponds to the way the human ear perceives sound, since its differentiation at higher frequencies only happens when the jump between them is greater.

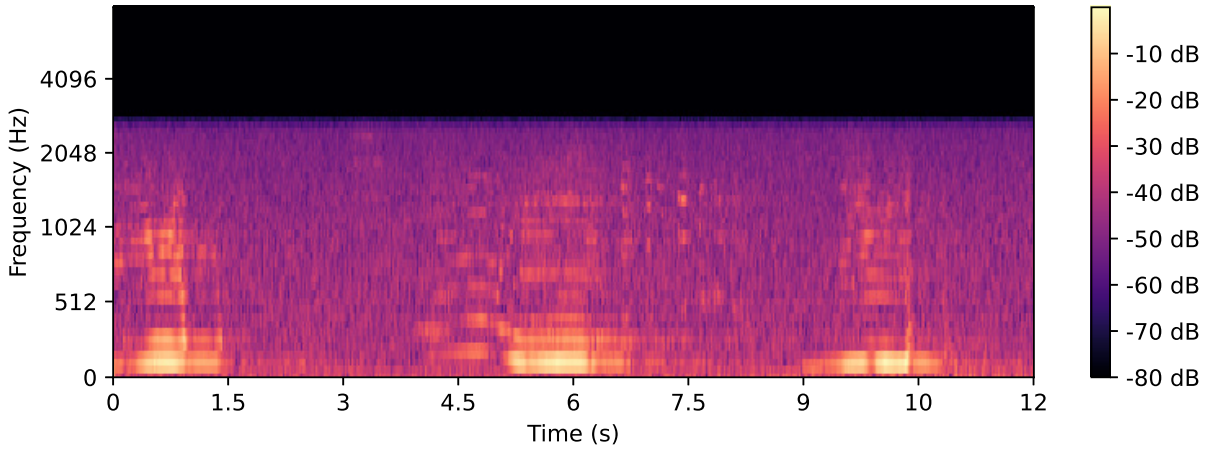

Figure 3.4: Mel spectrogram of a humpback whale vocalization

Chapter 4

Knowledge discovery process of recognition framework

The recognition framework was constructed based on the knowledge discovery process described in [19], which can be divided into four phases: a) Data selection, b) Feature extraction, c) Modeling and d) Evaluation. In Figure 4.1, the procedures adopted in each of the phases are shown. The feature extraction and the architecture of the proposed CNN were based on a thesis written by Nordby [16] and on [17]. Both use a similar CNN model to classify urban sounds and present a characterization of the feature extraction and the CNN architecture in a more detailed and straightforward manner. This is in contrast to other studies analysing marine sounds. Although these two studies do not deal with sounds in an aquatic environment, it is possible to draw a parallel between both urban and aquatic environments, due to a similar complexity of sounds.

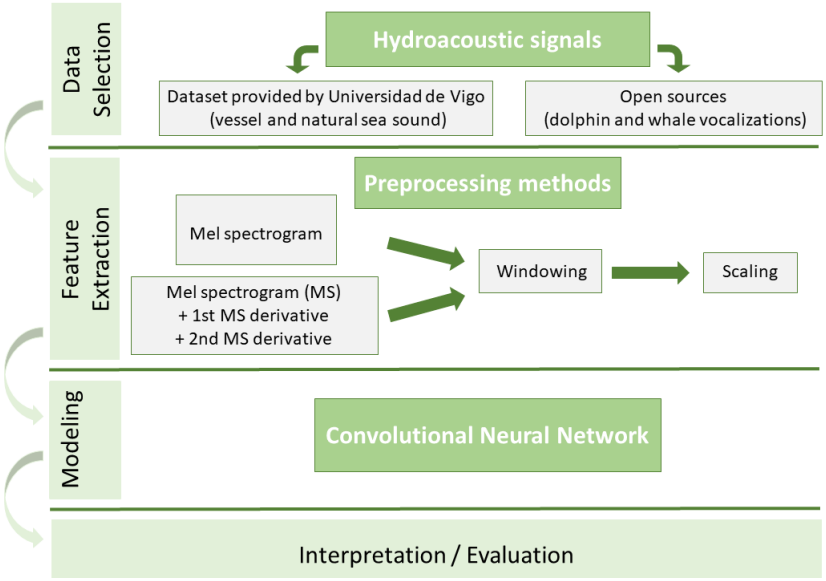

Figure 4.1: Knowledge discovery process of recognition framework

4.1 Data selection

In this work, the aim is to devise a model that analyses a hydroacoustic signal and classifies it according to the sound sources presented in that signal. Above all, the objective is to classify all the possible types of marine sources in the ocean, such as cetaceans, fishes, surface vessels, submarines. As a database which incorporates a wide range of underwater signals is not available, it was decided to use only some of the main sound sources in the marine environment. Two main datasets have been used. One is provided by Universidad de Vigo, which is composed by vessel recordings and background noise (ShipsEar dataset [33]). The other complements the previous one by adding dolphin and whale vocalizations from different open sources, in order to spread the diversity of sound sources. These two datasets are described in detail in Chapter 5, where the several models based on the application of recognition framework to these datasets are shown.

In addition, in order to test the recognition framework, the developed methodology is applied to non-marine data (UrbanSound8k dataset [15]).

4.2 Feature extraction

According to [34], feature extraction can be split into two steps: *feature construction* and *feature selection*. The former aims to improve the quality of the data since in the real world data usually has irrelevant or redundant information. The most common methods consist in removing noise from a signal, dealing with missing values, normalizing, or standardizing the dataset. Thus, *features construction* involves converting "raw" data into significant features and it is usually performed by preprocessing methods. The *feature selection* chooses the most relevant features which will have a higher influence on the desired outcomes of the model. This leads to a data dimensional reduction that reduces storage requirements, increases the algorithm speed, and improves prediction accuracy. This latter is generally due to overfitting reduction.

In the recognition framework, *feature construction* includes three preprocessing transformations: 1) Conversion to mel spectrogram, 2) Windowing and 3) Scaling. Another similar approach is also applied, in which, besides the three aforementioned steps, the calculation of the first and second mel spectrogram derivatives is included. Therefore, two different feature extraction methods are part of the recognition framework.

There is one more preprocessing transformation, which is integrated into the modeling process. As explained in Subchapter 2.2, convolutional layers are formed by filters that extract knowledge from the input and decrease its dimension. In this way, specific knowledge is transformed into features which will be used as inputs in a group of fully-connected layers (the last part of any ConvNet). Thus, *feature selection* is not taken into account, since the last preprocessing method is inserted into the modeling process.

4.2.1 Conversion to mel spectrogram

The first preprocessing method consists in converting the digital signal into a mel spectrogram. Firstly, the STFT is applied in order to compute the spectrogram, and, secondly, the spectrogram frequency scale is converted to a mel scale.

When calculating the mel spectrogram, some parameters have to be defined from the beginning. In the case of the STFT, the signal is framed using the *Hann window* with a length of $M = 2048\ samples$ (39 ms at 52734 Hz) and a hop between frames of $R = 1024\ samples$ (19 ms at 52734 Hz). In the conversion of the frequency scale to mel scale, the number of filter banks is set to 60 bands. Thus, the mel spectrogram is a matrix of size $(60,\ \#frames)$. Finally, the mel spectrogram coefficients are converted to decibels.

The mel spectrogram is produced using the *Librosa* library [35].

4.2.2 First and second mel spectrogram derivatives

The feature extraction can be expanded by adding the first and second derivatives of the mel spectrogram, in order to incorporate information about the dynamic behaviour of the parameters over the frequency axis. This process is based on [36], which combined the cepstral coefficients and their first and second derivatives as a feature vector of a Gaussian mixture model in order to classify dolphin vocalizations.

The derivatives are computed from the difference between the mel spectrogram coefficients (c), before being converted to decibels, along the frequency axis. As a 60 mel scale filter bank is used, the first derivative can be determined as follows:

$$\begin{cases} d_i = \frac{c_{i+1} - c_i}{2}, & i = 1, \\ d_i = \frac{c_{i+1} - c_{i-1}}{2}, & 2 \leq i \leq 59, \\ d_i = \frac{c_i - c_{i-1}}{2}, & i = 60. \end{cases} \tag{14}$$

This process is repeated for each frame, creating a matrix with the first derivative values, which has the same size as the mel spectrogram matrix. The second derivative is also computed using the previous formula, but applied to the first derivative matrix rather than to the mel spectrogram coefficients.

Thus, the mel spectrogram and its first and second derivatives form a three-dimensional matrix of size $(60,\ \#frames,\ 3)$.

4.2.3 Windowing

Large signals are usually difficult to analyze, as they can contain too much information, which is an obstacle to discovering patterns that would allow the model to distinguish between classes. Therefore it is

more practical to divide each signal into shorter segments. This leads to the next preprocessing method: windowing.

In this process, a *rectangular window function* with a specific length in frames is applied, creating a set of windows with the same size for a complete recording. An example of the application of this windowing technique is shown in Figure 4.2. In this case, the duration of each window is 1.00 s, which following the reasoning of Equation 11, corresponds approximately to 50 frames when $sr = 52734\ Hz$, $M = 2048\ samples$ and $R = 1024\ samples$:

$$window\ length\ (frames) = \frac{window\ length\ (seconds) \times sr - O}{R} = \frac{1.00 \times 52374 - 1024}{1024} \approx 50, \tag{15}$$

where $O = M - R$.

Although the *rectangular window function* is applied to the entire mel spectrogram without superimposition with respect to frames, we should note that consecutive frames overlap in time, so the first frame of a given window overlaps partially with the last frame of the previous window. For instance, in Figure 4.2, the second window, which starts at frame 51, still contains a small amount of information about the signal before 1.00 s.

Windowing process is also applied to the first and second derivative matrices in the same way as with the mel spectrogram.

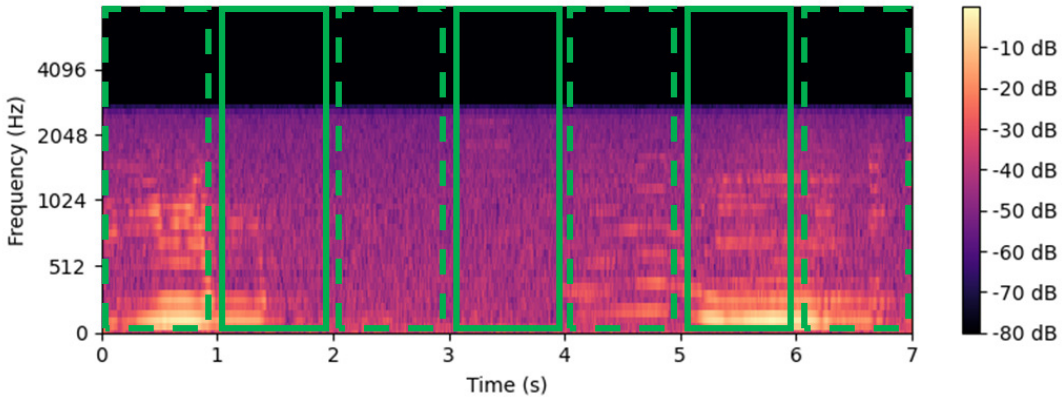

Figure 4.2: Windowing

4.2.4 Scaling

After splitting the features into windows of the same size, a feature *scaling* process is applied to each window, in order to bring all the features to the same level of magnitude. The two most common scaling processes are normalization and standardization. In this work, standardization is used as the scaling process, which is applied by:

$$c_i{}' = \frac{c_i - mean(c)}{stdev(c)}. \tag{16}$$

Standardization changes the mean and standard deviation of the features to 0 and 1, respectively, and since it does not have boundaries, outliers do not have a negative impact on scaling.

When each window is composed by a three-dimensional matrix (mel spectrogram and first and second derivatives), scaling is applied to each one separately.

4.3 Modeling

In feature extraction, the acoustic signal is converted to a mel spectrogram representation or a mel spectrogram representation and its first and second derivatives, depending on the chosen approach. Regardless of the input dimensions, the same CNN was applied.

The CNN architecture consists in three convolutional layers followed by two fully connected layers. Each convolutional layer has a kernel size of 5, where the first convolutional layer uses 24 kernels and the second and the third convolutional layers use 48 kernels. In the three convolutional layers and in the first fully connected layer, ReLU is used as an activation function. For the output layer, the typical activation function in multiclass classification is used: softmax function. The size of the fully connected output layer takes into account the number of classes.

In addition, there are two max-pooling layers interspersed among the three convolutional layers, with a filter size and a stride depending on the size of the input window. Batch normalization is performed on each convolutional layer. With regard to the two fully connected layers, in order to avoid overfitting, dropout and L2-regularization are applied. The former uses probability 0.5, meaning that one in two inputs will be randomly excluded from each update cycle. The latter is applied to the weights with a penalty factor of 0.001. Cross-entropy function is used as loss function and gradient descent with Nesterov momentum set to 0.9 is used to perform optimization.

The architecture of the proposed CNN model is represented in Figure 4.3.

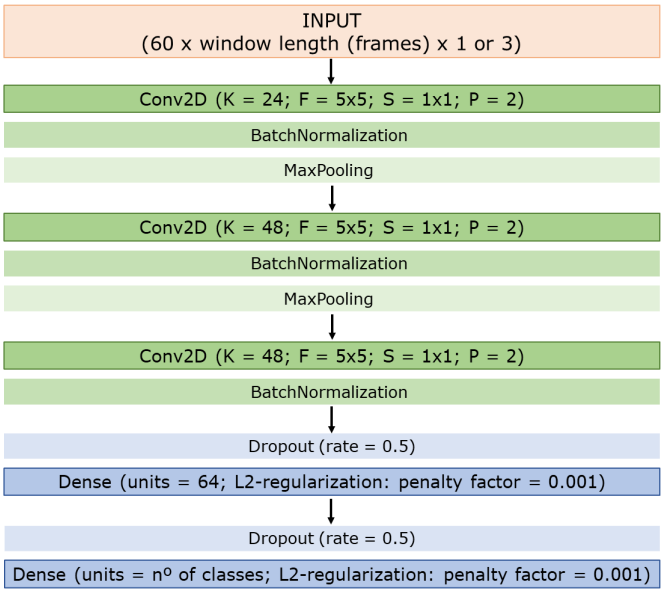

Figure 4.3: Architecture of the CNN

Therefore, the CNN receives windows with the mel spectrogram representation, or the mel spectrogram with the addition of the first and second derivatives as input, and it assigns class scores to each window. The class with the highest score is the class predicted for the specific window. In this way, the classifier makes predictions for each individual window instead of doing a classification for the signal as a whole. CNN is built using Keras library with TensorFlow.

4.4 Evaluation

Four metrics are used to evaluate the performance of the classifiers. These are based on indicators for the multi-class classification reviewed in [37]. The accuracy corresponds to the percentage of correct classifications for the entire set of testing data.

$$Accuracy = \frac{Total\ number\ of\ correct\ classifications}{Total\ number\ of\ windows}. \tag{17}$$

Precision and recall for each class are computed considering true positives as the only correctly classified windows for the class under study. Precision expresses the probability of a prediction being correct for that specific class, which quantifies how well the classifier prevents false positives. Recall measures the accuracy for the class under study.

$$Precision_{class} = \frac{True\ positives\ (class)}{Total\ number\ of\ predicted\ positives\ (class)}. \tag{18}$$

$$Recall_{class} = \frac{True\ positives\ (class)}{Total\ number\ of\ actual\ positives\ (class)}. \tag{19}$$

The F1-score is computed as the harmonic mean between precision and recall. It is used to combine these two metrics into a single one, which can be interpreted as a weighted average between both.

$$F1\text{-}score_{class} = 2 \cdot \left(\frac{Precision_{class} \cdot Recall_{class}}{Precision_{class} + Recall_{class}} \right). \tag{20}$$

4.5 Testing the framework on urban sounds

In machine learning models, data quality has a huge impact on its performance. This kind of models need to be trained with a set of high-quality data, *i.e*, a dataset which correctly represents the total problem domain, in order to get accurate predictions. In addition, the size of the dataset also has a great relevance. Generally, large datasets lead to higher accuracy.

The marine environment is extremely complex, where there are diverse types of sound sources and each source can produce significantly different sounds depending on the situation. In addition, as explained in Chapter 3, the temperature, salinity and pressure have a high influence in the propagation of sound waves, and the water depth at which the hydrophones are located also has an impact on the characteristics of the recorded sounds. So, a high number of samples for each class, covering a varied type of situations is needed. However, these are difficult to obtain in the scope of a Msc. work, there is not much data freely accessible. Even embracing some techniques to increase the amount of data (data augmentation), this might not be enough to acquire the number of data which machine learning models demand. Thus, in order to have more confidence when extending the dataset size in the future, finding a way to test the recognition framework is necessary.

Therefore, it was decided that it would be important to test the devised methodology for a large dataset with non-marine sounds, which could be adaptable to this work. To this end, an open-source dataset composed of urban sounds (UrbanSound8K dataset [15]) was used as a benchmark. This dataset is constituted by ten classes of sounds that can be heard in a city environment. Although the sounds are not in an aquatic environment, they correspond to sound sources considerably different from each other and as complex as the marine sounds. Additionally, another advantage to using this dataset is the high number of published studies which use this dataset, including the two studies that the recognition framework was based on.

Five models were constructed using as data the UrbanSound8K dataset. Below, all the work developed using the urban sounds dataset is explained. First, the characteristics of this dataset are detailed and the implementation of the recognition framework is described. Then, results are presented and discussed.

4.5.1 UrbanSound8K dataset description

UrbanSound8K is a dataset of urban sounds with a total of 8732 labeled recordings and each one is labeled to one of ten distinguished classes [15]. These are: Air Conditioner (AC), Car Horn (CH), Children Playing (CP), Dog Bark (DB), Drilling (Dri), Enginge Idling (EI), Gun Shot (GS), Jackhammer (Jac), Siren (Sir) and Street Music (SM). The duration of each recording is up to 4.00 s.

The recordings are distributed in 10 folders (Fold1-Fold10). In a single folder, there are excerpts from the same recording, but this never happens between folders. Table 4.1 shows information about the class distribution in each folder and the overall percentage for each class, in order to understand the balancing of the dataset. There are two classes (Car Horn and Gun Shot) which have less than a half the data in comparison to the other eight, meaning that the dataset is not totally balanced.

Folder	AC	CH	CP	DB	Dri	EI	GS	Jac	Sir	SM
1	100	36	100	100	100	96	35	120	86	100
2	100	42	100	100	100	100	35	120	91	100
3	100	43	100	100	100	107	36	120	119	100
4	100	59	100	100	100	107	38	120	166	100
5	100	98	100	100	100	107	40	120	71	100
6	100	28	100	100	100	107	46	68	74	100
7	100	28	100	100	100	106	51	76	77	100
8	100	30	100	100	100	88	30	78	80	100
9	100	32	100	100	100	89	31	82	82	100
10	100	33	100	100	100	93	32	96	83	100
Total	1000	429	1000	1000	1000	1000	374	1000	929	1000
%	0.1145	0.0491	0.1145	0.1145	0.1145	0.1145	0.0428	0.1145	0.1064	0.1145

Table 4.1: Class distribution of UrbanSound8k dataset

Mel spectrogram example for each class of the urban sound database is displayed in Figure 4.4. The frequency ranges between 0 Hz and 18000 Hz.

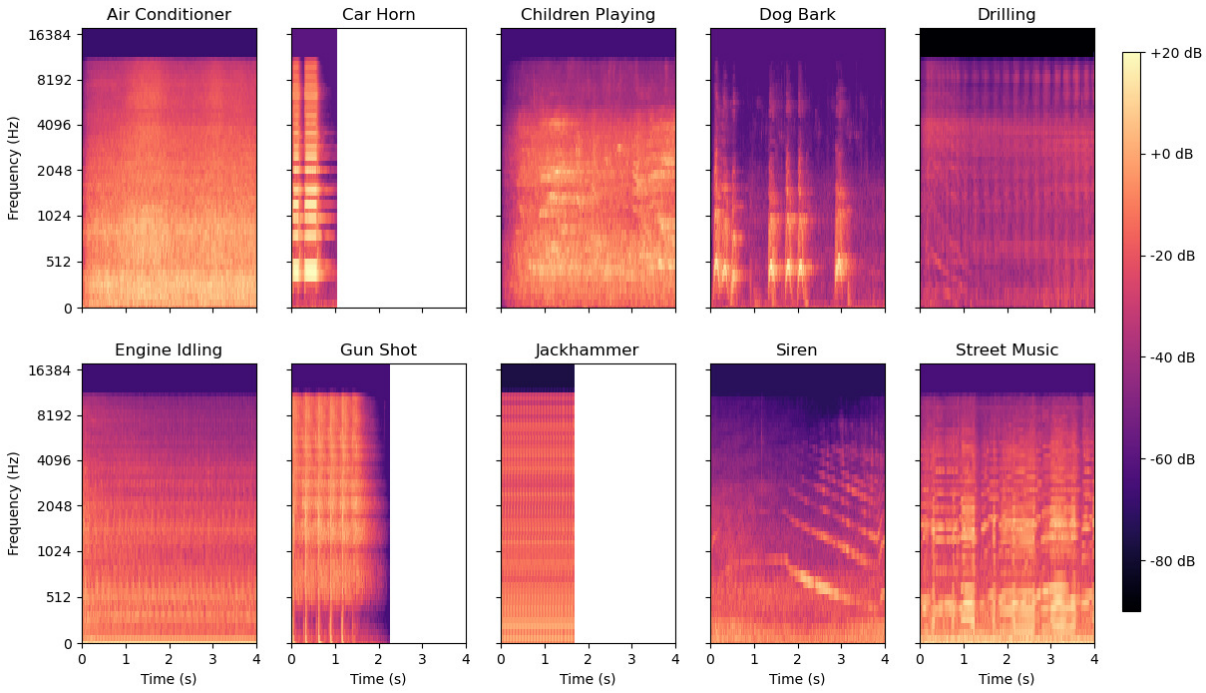

Figure 4.4: Mel spectrogram class representation of UrbanSound8k datset

4.5.2 Parametrization

In the UrbanSound8k dataset, only one of the feature extraction methods was used, which consisting solely in converting the acoustic signal into the mel spectrogram. The sampling rate of each recording is highly variable. The most common value is 44100 Hz. All the data was resampled to 22050 Hz, since this still keeps the quality of the recording, while decreasing the computational time, which is important given the size of the dataset, keeping it in line with Nordby's work [16].

A different sampling rate from the default rate (52734 Hz) leads to a few adjustments to the parameters of the mel spectrogram. Thus, for the present dataset, regarding the parameters related with STFT, the signals were framed using a *Hann window* with a length of $M = 1024\ samples$ (46 ms at 22050 Hz) and a hop between frames of $R = 512\ samples$ (23 ms at 22050 Hz). The frequency range was set to 0 - 8000 Hz, since this was enough to include the majority of relevant frequencies, as shown in Figure 4.4. The set parameters are shown in Table 4.2.

In total, five similar models were built, with the difference being in the windowing process. A *rectangular window function* of different size was applied to each model. This was in order to understand the impact of size on the performance of the model. As the maximum occurrence duration for each recording is limited to 4.00 s, a *rectangular window function* of approximately **0.75 s**, **1.42 s**, **2.21 s**, **3.00 s** and **4.00 s** was applied, which, following the reasoning of Equation 15, corresponds to **31 frames**, **60 frames**, **94 frames**,

128 frames and **171 frames**, respectively. For recordings that have a duration shorter than the rectangular window length, the mel spectrogram was extended applying zero-padding until the rectangular window length was reached so as to insure at least one window for each recording. Each model was trained for up to 50 epochs and with a batch size of 200.

Sampling rate	22050 Hz
Window function	Hann window
M	1024 samples (46 ms)
R	512 samples (23 ms)
Filter banks	60 bands
Minimum frequency	0 Hz
Maximum frequency	8000 Hz

Table 4.2: Parameters for UrbanSound8k dataset

The CNN had to be adapted taking into account the number of frames of the input, more precisely the two max-pooling layers. The dimensions of its filter size and stride were adapted to the dimension of each input as follows:

Input dimension	(60 x 31 x 1)	(60 x 60 x 1)	(60 x 94 x 1)	(60 x 128 x 1)	(60 x 171 x 1)
Filter size	3 x 2	3 x 3	3 x 3	3 x 4	3 x 5
Stride	3 x 2	3 x 3	3 x 3	3 x 4	3 x 5

Table 4.3: Max-pooling layer parameters for UrbanSound8k dataset

4.5.3 Results

To evaluate and compare the performance of the different classification models (which differ in window length), each model was run using a 10-fold cross-validation [38]. This methodology is recommended in the presentation document of the present dataset [15], and it allows a direct comparison of the present results with those of other works. One of the nine training folds in each split was randomly selected as validation set for identifying the training epoch that yields the best model parameters when training with the remaining eight folds. To this is end, the identification was based on the validation accuracy

Since the classifier runs on predictions per window, an evaluation for each recording was set by combining the window predictions. So, based on the average score of each class for all windows of each recording, the one with the highest average was the class predicted for the recording. This procedure is called *soft voting* and it is usually used when there is a group of classifiers and the global prediction computation is required [39]. Thus, in Equation 17, the number of windows becomes the number of recordings. The same procedure was used by Nordby [16].

The results of the five models are reported as a box-and-whisker plot generated from the accuracy scores of the 10 folds, as shown in Figure 4.5. The cross and horizontal line represent the average and median

accuracy, respectively. All models showed a stable performance with a very slight improvement when increasing the window length. The average accuracy of each model with the increase in the window length was 69.3%, 70.4%, 70.6%, 71.1% and 72.2%, respectively. These results can be compared to [16], which proposed a CNN model using mel spectrogram windows of 0.72 s as input, and to [17], which proposed the same approach but with windows of 3.00 s. The best performance of the two studies without data augmentation was 72.3% and 73%, respectively, which are slightly higher than the accuracy of the present work. Nevertheless, it is possible to conclude that the performance of the present models are actually very close to the expected range. This gives confidence regarding the devised recognition framework.

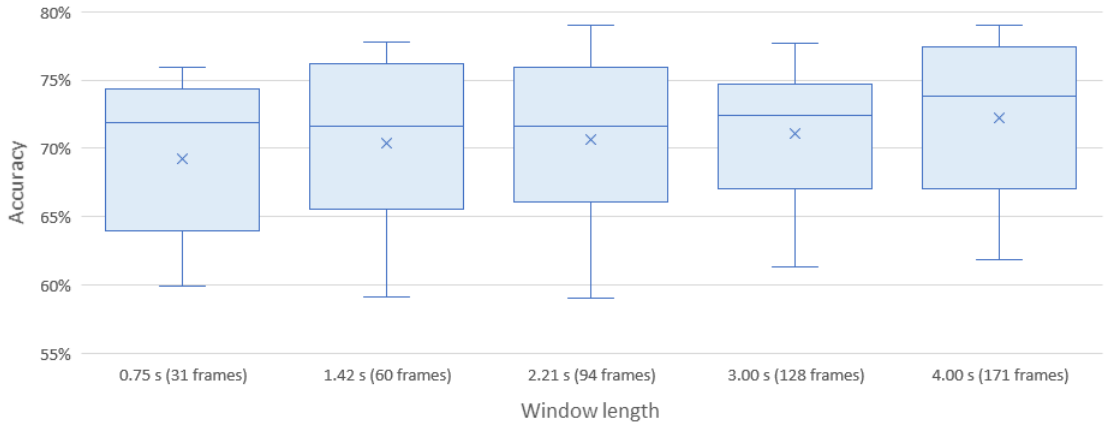

Figure 4.5: Test accuracy for the five models using different window length (UrbanSound8k dataset)

Chapter 5

Application of the recognition framework in the marine environment

Once the designed methodology for recognizing acoustic signals provided the expected result in the recognition framework testing process, the methodology is now applied to aquatic environment sounds. For this purpose, two datasets are used. The first is mostly composed by vessel recordings which vary from small fishing boats to large vessels designed to carry cars (ShipsEar dataset). A few natural noise recordings are also part of this dataset. The second dataset was built by us, and its purpose is to evaluate the capabilities of the model into a closer approximation to what could be recorded in the ocean, including vessel noises, natural noises and animal sounds.

5.1 Sounds produced by vessels

ShipsEar is a dataset composed of vessel sound recordings (available at `http://atlanttic.uvigo.es/underwaternoise/`), and it was introduced by Santos-Domínguez et al. [33]. The same authors also developed a vessel classifier for the same dataset based on the cepstral coefficients plus its first and second derivatives and the Gaussian mixture model.

Since 2016, further studies with different approaches have been published using the ShipsEar dataset. Two studies published in 2020 (Li et al. [40]; Ke et al. [41]) are noteworthy. Li et al. [40] proposed a classic approach of ASR. The feature extraction was based on the filter banks and the mel frequency cepstral coefficients (MFCCs), using as classifier a DNN. However, an additional process was introduced to the classic methodology. Thus, after extracting features from the acoustic signal, there was an optimization process based on the triplet loss concept in order to increase the inter-class distance and reduce the intra-class distance. On the other hand, Ke et al. [41] paid more attention to the feature extraction, computing different kinds of features. In total, eight groups of features were extracted from the acoustic signal: temporal, statistical, spectral, cepstral, Hilbert spectral, wavelet and deep neural network features. The k-nearest neighbour was used as classifier method.

5.1.1 Dataset description

The ShipsEar data recordings were collected in different locations of the Spanish Atlantic coast with a maximum of three hydrophones. The majority of recordings were made in Vigo Ria, near the port of Vigo, but also included a few excerpts from La Coruña outer harbour. Apart from the vessel sounds, different types of natural background noises (wind, rain, waves and current) are also part of this dataset. For these natural noises, the hydrophones were located far from traffic routes, outside Vigo Ria, in order to minimize sounds generated by vessels.

The port of Vigo is one of the busiest ports in Europe in terms of passengers and goods. In the ShipsEar dataset, there are recordings of variety of vessels ranging from simple fishing boats or tugboats to large vessels, such as ocean liners, which transport passengers across oceans, or even ro-ro vessels, which are designed to carry cars or trucks. Besides these, there are recordings of passenger ferries and leisure boats (motorboats, pilot boats, sailboats).

In total, the ShipsEar dataset is made up of 90 recordings, in which 11 types of vessels are identified. The duration of each recording is between 15 s and 10 min. According to [33], the 11 vessel types plus the natural noises were divided into five classes. Four classes were allocated to vessel sounds and one to natural sea noises, as shown in Table 5.1. Vessels were classified according to size. In the present work, the same class distribution was used.

Classes	Recording sources	Recordings (nº)
Class A	fishing boats, trawlers, mussel boats, tugboats, dredgers	17
Class B	motorboats, pilot boats, sailboats	19
Class C	passenger ferries	30
Class D	ocean liners, ro-ro vessels	12
Class E	background noise recordings	12

Table 5.1: Classes of ShipsEar dataset

The number of recordings is unevenly distributed when considering class. In Figure 5.1, which shows the time duration for each class, these difference is even more mark when looking at the data collected.

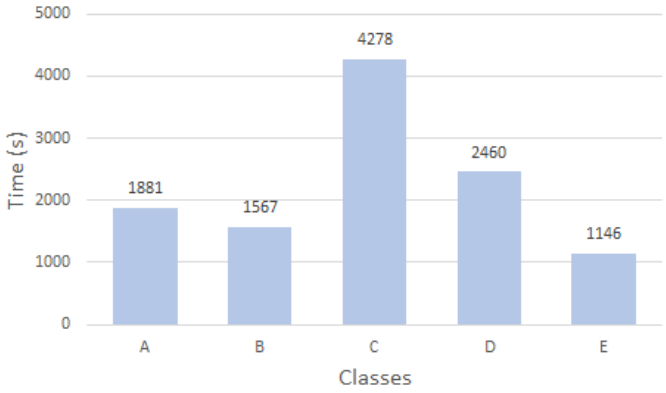

Figure 5.1: Class distribution (ShipsEar dataset)

Santos-Dominguez et al. [33] explained that "*The recordings were segmented with wide margins to preserve information from the beginning to the end of the event*". However, using these margins would have had negative effects on the current methodology. The developed classifier runs on predictions per window along the signal, as opposed to doing a single classification for the whole signal. In order to be absolutely certain that all windows are labeled correctly, those margins were excluded, working only with the relevant intervals, *i.e*, those containing vessel sounds. Therefore, an interval for each recording was defined discarding these margins. The intervals selected are presented in Appendix A, which also took into account the difference in the data collected, according to the number of recordings and the time duration, in order to to avoid large differences in the class distribution. For example, smaller intervals for recordings of class C were chosen, which had the greatest amount of recordings.

In addition, five recordings were excluded (identified in grey in Appendix A) that were considered as outliers, either because the vessels were not moving (three from class C and one from class D) or because the vessel signal occurred for too short time and with a low amplitude (one from class B).

It was decided to reduce the intervals of some recordings, rather than exclude them, as had been done with the outliers. This was important, since recordings from the same vessel type, containing different characteristics, lead to a greater diversity in the database, making it more reliable.

The final total of recordings was **85 recordings**. The final class distribution is shown in Figure 5.2.

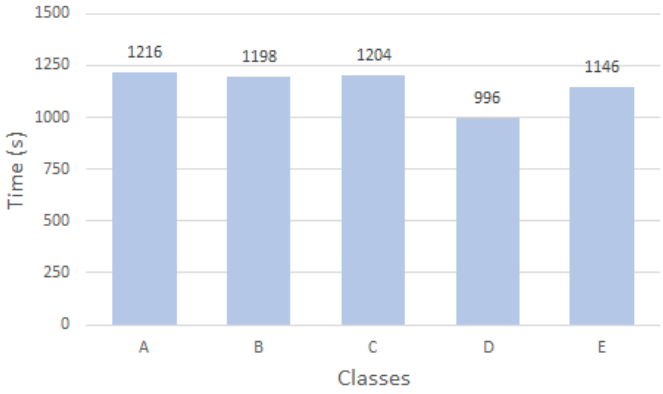

Figure 5.2: Final class distribution (ShipsEar dataset)

In Figure 5.3, some mel spectrogram representations from the ShipsEar dataset are shown. Each class is represented by intervals of 10 s from two different recordings: class A (tugboat and mussel boat), class B (motorboat and pilot boat), class C (two passenger ferry recordings), class D (ro-ro vessel and ocean liner) and class E (two background noise recordings). By comparing the mel spectrogram representations of each class, it is not so easy to find patterns that distinguish different types of vessels according to their size. However, there is a clear difference between the vessel sounds and the background noise, since the former are characterized by a higher and constant energy in low frequencies, which are represented by horizontal lines near the bottom of the mel spectrogram, whereas the background noise is not represented by any particular frequency.

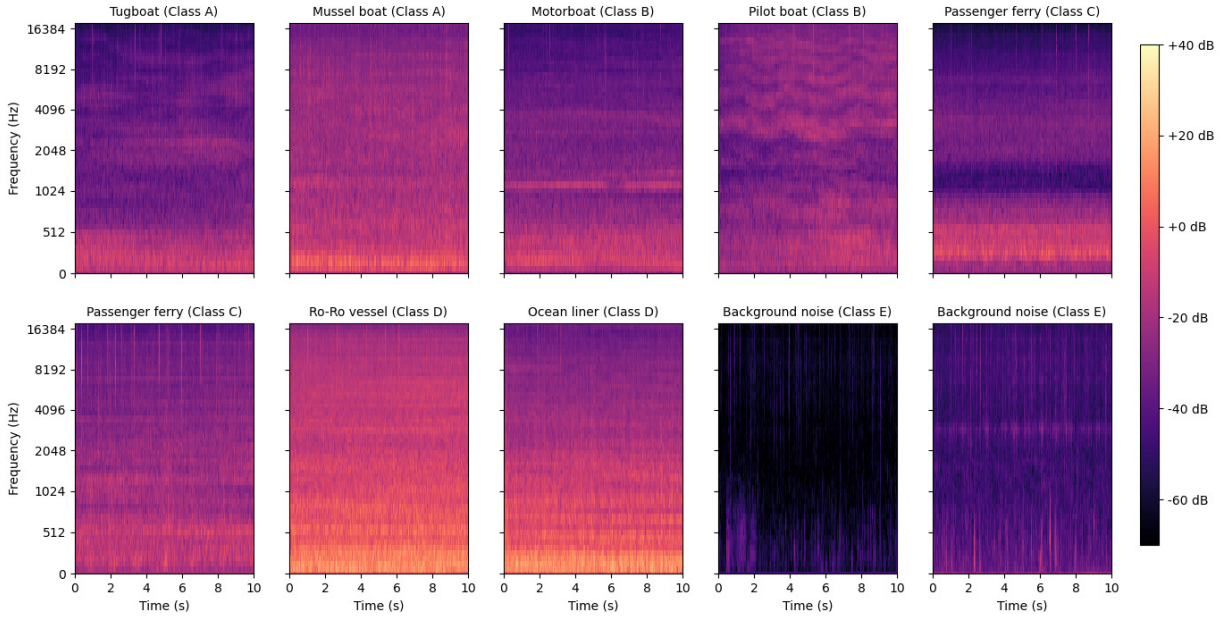

Figure 5.3: Mel spectrogram representations of each class from the ShipsEar dataset

5.1.2 Parametrization

The original sampling rate of all recordings was kept at 52734 Hz. The signal was framed using a *Hann window* with a length of $M = 2048\ samples$ (39 ms at 52734 Hz) and a hop between frames of $R = 1024\ samples$ (19 ms at 52734 Hz). The mel spctrogram was computed with 60 bands. The frequency range was set between 0 Hz and 8000 Hz, because most of the relevant frequencies are below 8000 Hz. The set parameters are shown in Table 5.2.

Sampling rate	52734 Hz
Window function	Hann window
M	39 ms (2048 samples)
R	19 ms (1024 samples)
Filter banks	60 bands
Minimum frequency	0 Hz
Maximum frequency	8000 Hz

Table 5.2: Parameters for ShipsEar dataset

For this dataset, two models were built using the mel spectrogram and the mel spectrogram and its first and second derivatives as features. In contrast to the analysis carried out for the UrbanSound 8k dataset, where it was studied in detail how the window length, in particular the number of frames, affects the performance, now the same number of frames per window was used and the focus was understanding the impact of different features. In this case, the window length was set approximately to **1.00 s**, which according to Equation 15 corresponds to **50 frames**. Thus, the input dimension of the CNN was (60 x 50 x 1) or, if the mel spectrogram is coupled with its first and second derivatives, (60 x 50 x 3). The same CNN architecture was used in both cases, where the two max pooling layers were defined with a filter size of 3x3 and stride 3.

5.1.3 Data splitting

Each model was built and evaluated using a 10-fold cross-validation [38]. The same process was also used in [41]. In data splitting, there was random sampling of windows for training and testing data. In each split, 10% of training data was used as validation data in order to identify the training epoch that yields the best model parameters based on validation accuracy. Each model was trained for up to 50 epochs and with a batch size of 200.

5.1.4 Results

In Figure 5.4, an overview of the performance for the two models with different feature extraction is shown as a box-and-whisker plot of the ten estimates of the accuracy (see Equation 17) generated from the cross-

validation. The cross and horizontal line represent the average and median accuracy, respectively. The feature combination between the mel spectrogram and the first and second derivatives performed better than when only using the mel spectrogram. The average accuracy of the two models was 88.8% and 83.2%, respectively.

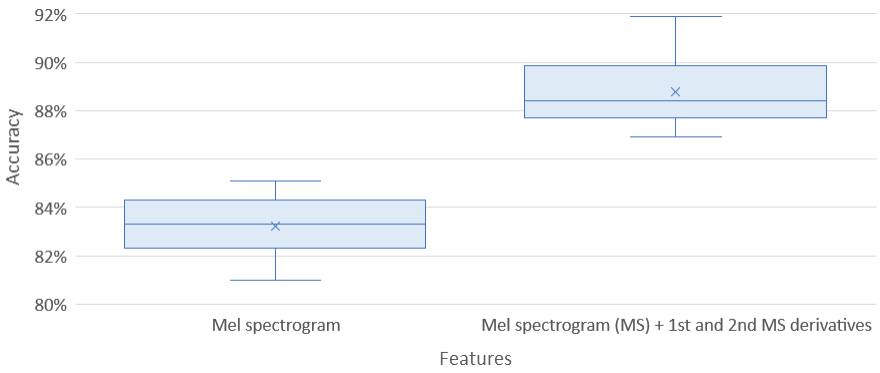

Figure 5.4: Test accuracy of the two models with different feature extraction

The precision, recall and F1-score metrics (see Equations 18-20) are used to evaluate the performance for each class individually. The results are shown in Table 5.3. Taking into account the F1-score, which is a weighted average between precision and recall, it confirms that the feature combination between the mel spectrogram and its derivatives performed better than solely using the mel spectrogram as features. The F1-score shows an improvement for all classes, especially for vessel classes, where this metric increased between 6.2% and 7.1%. Class E recorded the lowest improvement, yet the F1-score was already extremely high when using the mel spectrogram as features.

The precision and recall of class E indicates that there is a reliable identification of the presence and absence of vessel noise for both models. The precision shows that most of the samples classified as class E were correct, and the recall indicates that the majority of class E data was predicted accurately. Any misclassifications in the four vessel classes were, thus, for the most part, between these four classes.

Metric	Features	Class				
		A	B	C	D	E
Precision (%)	Mel spectrogram (MS)	78.7	83.1	79.2	77.5	97.5
	MS + 1st and 2nd MS derivatives	81.8	91.4	85.4	87.1	98.6
Recall (%)	Mel spectrogram (MS)	74.4	82.9	75.4	87.1	97.8
	MS + 1st and 2nd MS derivatives	85.3	87.3	81.6	91.2	99.5
F1-score (%)	Mel spectrogram (MS)	76.5	83.0	77.2	82.0	97.7
	MS + 1st and 2nd MS derivatives	83.5	89.3	83.4	89.1	99.1

Table 5.3: Classifier precision, recall and F1-score of the two feature extractions

The confusion matrices of the two models are shown in Tables 5.4 and 5.5, which indicate the percentage of correct predictions for each class along the diagonal (recall values), and the wrong classifications elsewhere. The values in brackets correspond to the conversion of the percentage values into the number of samples (windows). The confusion matrices confirm that both models achieved a good performance in the perception of the presence and absence of vessel noise, where the most of the misclassifications occurred in distinguishing the vessels according to size.

Although there are a few studies, already mentioned above, which used the same dataset to develop a classifier to recognize hydroacoustic signals, there is no correlation between the present results and those of the other studies. This is due to the lack of uniformity in the data used to build and evaluate each model because, albeit being the same dataset, there is no detailed information about the recordings used and the data splitting process.

		Predicted class				
		A	B	C	D	E
True class	A	74.4% (925)	5.7% (70)	9.3% (114)	13.0% (133)	0.1% (1)
	B	5.0% (62)	82.9% (1016)	7.1% (87)	4.2% (43)	1.5% (17)
	C	8.8% (110)	8.8% (108)	75.4% (928)	7.7% (79)	0.5% (6)
	D	5.6% (69)	1.8% (22)	2.9% (36)	87.1% (888)	0.4% (5)
	E	0.7% (9)	0.6% (7)	0.6% (7)	0.3% (3)	97.8% (1146)

Table 5.4: Confusion matrix using as features the mel spectrogram

		Predicted class				
		A	B	C	D	E
True class	A	85.3% (1060)	2.8% (34)	6.2% (76)	7.1% (72)	0.1% (1)
	B	5.3% (66)	87.3% (1069)	5.5% (68)	1.3% (13)	0.8% (9)
	C	8.8% (110)	4.7% (58)	81.6% (1004)	5.2% (53)	0.5% (6)
	D	4.7% (58)	0.6% (7)	2.0% (25)	91.2% (930)	0% (0)
	E	0.2% (2)	0.1% (1)	0.2% (3)	0.0% (0)	99.5% (1166)

Table 5.5: Confusion matrix using as features the mel spectrogram and the first and second derivatives

5.2 Sounds produced by marine animals and vessels

In this second stage, the spectrum of sound sources was spread. In addition to the vessel and natural sea noise of the ShipsEar dataset, two marine animal vocalization sources were integrated into the dataset. The ideal scenario would be to add recordings of several species of marine animals to the ShipsEar data. However, these are difficult to obtain during this type of work. So, the focused was on collecting data from only two marine animals: dolphins (common dolphin, striped dolphin and bottlenose dolphin) and humpback whales, these species being very common in the Atlantic Ocean. Therefore, this dataset was used to study the capacity of the methodology in a scenario of vessel noise and marine animal vocalizations, and to analyse specific cases in detail, looking at a real-time application of the model based on the window results.

5.2.1 Dataset description

In total, five classes were used: **dolphins**, **humpback whales**, **small vessels**, **large vessels** and **background noise**. The dolphin class covers recordings of three different species: common dolphin, striped dolphin and bottlenose dolphin. The marine animal data was collected from open sources. All the information about the source of each recording is available in Appendix B. In addition, two recordings of common dolphin sounds provided by WavEC were added to the dolphin data.

For the vessel class, data from the ShipsEar dataset was used. In the first application of this dataset, the data was divided into five classes (four for vessel sounds and one for background noise). The vessel recordings were distributed into four classes according to vessel size. Now, the same criteria was followed in this study. However, the four classes were merged into just two. Class A and B were merged into **small vessels** class and class C and D into **large vessels** class. Since the data sample of marine animals was very small, parts of the ShipsEar recordings have to be selected in order to avoid large differences in the class distribution. Thus, from the 85 recordings used in the application of the recognition framework for vessel sounds, 11 recordings from each class (A, B, C, D) were selected with an interval of approximately 20 s each. Regarding the background noise, the data once again came from the ShispEar dataset. All the background noise recordings (12) with an interval of approximately 40 s each were used.

All the data used from the ShipsEar dataset is listed in Appendix A in yellow. In Table 5.6, the data is summarized with information about the duration and the sampling rate of each recording. As can be seen, the sampling rate is quite variable. Thus, it was decided to set it into 52734 Hz, in order to keep the ShipsEar dataset sampling rate. In a few cases of dolphin and humpback whale data, the duration of the recordings is lower than that available on the original recordings, since intervals without vocalizations were excluded.

Dolphins	Humpback Whales	Small vessels	Large vessels	Background noise
Dolphin1.wav (7.6s;44100Hz) Dolphin2.wav (6.9s;44100Hz) Dolphin3.wav (9.4s;44100Hz) Dolphin4.wav (7.9s;22050Hz) Dolphin5.wav (49.7s;44100Hz) DolphinWavEC1.wav (37.3s;192000Hz) DolphinWavEC2.wav (36s;192000Hz)	Whale1.wav (11s;22050Hz) Whale2.wav (10.1s;22050Hz) Whale3.wav (10.6s;22050Hz) Whale4.wav (12.1s;22050Hz) Whale5.wav (17s;44100Hz) Whale6.wav (26.2s;10000Hz) Whale7.wav (10.6s;8000Hz) Whale8.wav (29.5s;44100Hz)	22 intervals of $\approx$ 20 s extracted from Class A and B of ShipsEar dataset (52734Hz)	22 intervals of $\approx$ 20 s extracted from Class C and D of ShipsEar dataset (52734Hz)	12 intervals of $\approx$ 40 s extracted from Class E of ShipsEar dataset (52734Hz)
154.8 s (total)	127.1 s (total)	440 s (total)	439 s (total)	476 s (total)

Table 5.6: Class distribution

In Figure 5.5, some examples of the mel spectrogram representation from the marine animals data are represented, where frequency ranges from 0 to 18000 Hz. The two most common dolphin vocalizations are the whistles and the echolocation clicks. Whistles give rise to high frequency contours in the mel spectrogram (for example Figure 5.5 (a)), while the echolocation clicks create vertical line patterns. In Figure 5.5 (e), there is a mix of whistles and clicks. The humpback whale vocalizations are identified as arcs, which sometimes are close to horizontal bars, at low frequencies in the mel spectrogram.

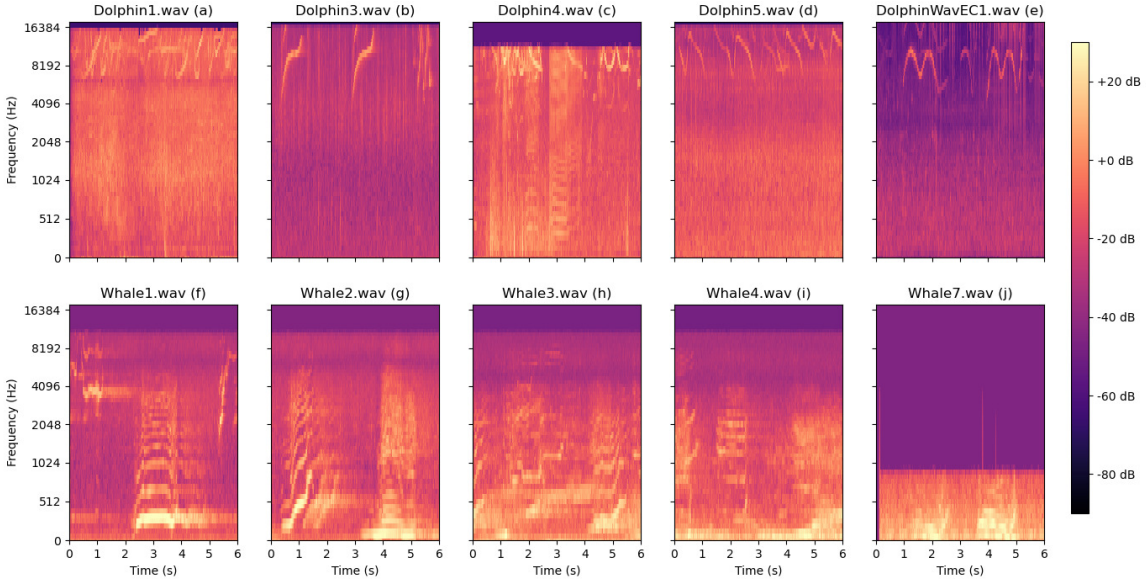

Figure 5.5: Mel spectrogram representations of dolphin and humpback whale vocalizations

5.2.2 Data augmentation

There is a significant difference in the amount of data between the two marine animal classes and the other three classes. In order to balance data, augmentation techniques were applied to dolphin and humpback whale data, which enlarge the dataset artificially. Data augmentation is quite common in the field of image classification [11][42], as well as in the the field of audio classification. Three common data

augmentation techniques based on [43][44] were applied to dolphin and humpback whale data, as follows:

- **Time stretching**: consists in speeding up and slowing down the signal without changing its pitch. Each recording of the dolphin and humpback whale classes was stretched by two factors: {0.8, 1.2}. If the factor is lower than 1, the signal is slowed down. If the factor is higher than 1, the signal is speeded up.
- **Pitch shifting**: consists in raising and lowering the pitch of the audio signal without changing its duration. Each recording of the dolphin and humpback whale classes was pitch shifted by two values in semitones: {-2, -4}.
- **Time shifting**: each recording of the dolphin and humpback whale classes is shifted over time, changing the original order of the signal. This was conducted by moving the last (or first, randomly selected) interval of each recording with a duration of $1/2$ and $3/4$ of the window length in seconds to the beginning (ending). In this way, the marine animal vocalizations will be placed in different positions in the mel spectrogram in contrast to the original signal.

Time stretching and pitch shifting were applied using the module *effects* of the *Librosa* library [35].

5.2.3 Parametrization

Only the feature combination of the mel spectrogram and the first and second derivatives was used here. As the original sampling rate of the ShispEar data was kept and the rest of the data was resampled to the same value, the mel spectrogram parameters correspond to the ones defined in Table 5.2. The only exception was the maximum frequency, which was set to 18000 Hz, since, as shown in Figure 5.5, dolphin vocalizations appear in the higher frequencies of the mel spectrogram.

Once again, the influence of the window length in the classification accuracy was studied in the same way as in the UrbanSound8K dataset. Five models were developed with a different window length in terms of the number of frames. Thus, a *rectangular window function* of a different size approximately **0.22 s**, **0.61 s**, **1.00 s**, **1.48 s** and **1.97 s** was applied to each model, which, taking into account Equation 15, corresponds to **10 frames**, **30 frames**, **50 frames**, **75 frames** and **100 frames**. Unlike the five models constructed for the UrbanSound8K dataset, where the window length was up to 4.00 s, in this case the window length did not go beyond 2.00 s, in order to make a more detailed analysis of smaller windows. This study was important since the objective is to develop a model that may have a future use as real-time classification.

Different dimensions in the windowing process requires an adaptation of the CNN architecture, which again was associated with the two max-pooling layers. The dimensions of its filter size and stride according to the size of the input are show in the following table:

Input dimension	(60 x 10 x 3)	(60 x 30 x 3)	(60 x 50 x 3)	(60 x 75 x 3)	(60 x 100 x 3)
Filter size	3 x 1	3 x 2	3 x 3	3 x 3	3 x 4
Stride	3 x 1	3 x 2	3 x 3	3 x 3	3 x 4

Table 5.7: Max-pooling layer parameters

5.2.4 Data splitting

A 3-fold cross-validation [38] at the level of the dolphin and humpback whale classes was used to build and evaluate each model. All recordings of these classes were divided into three subsets. This division was different to those used to the remaining classes, since the number of recordings of the other classes (2 classes of vessels and the background noise) is much larger and each recording is typically longer. Hence, in order to have a balanced testing data, only two recordings of the small vessels, large vessels and background noise classes were added to each one of the three subsets. Therefore, the evaluation was based on using one of the three subsets as testing data, with the remaining data being used for training (including the remaining recordings that were not added to the three subsets). The same process was repeated with each of the other two subsets as testing data. In this way, the sorting of the three subsets was such that insures that the same recording does not have some excerpts for training and some other for testing. Table 5.8 summarizes the division of the data into the three subsets.

	1st SUBSET		**2nd SUBSET**	
	Recording	Time (s)	Recording	Time (s)
Dolphin	Dolphin5.wav	49.7	DolphinWavEC1.wav	37.3
	DolphinWavEC2.wav	36	Dolphin1.wav	7.6
Humpback whale	Whale1.wav	11	Whale2.wav	10.1
	Whale3.wav	10.6	Whale6.wav	26.2
	Whale4.wav	12.1	Whale7.wav	10.6
Small vessels	94__A__Draga_2.wav	20	96__A__Draga_4.wav	20
	21__18_07_13_lanchaMotora.wav	20	45__19_07_13_yate_Sale.wav	20
Large vessels	10__10_07_13_marDeOnza_Sale.wav	20	40__19_07_13_MarDeCangas_Llega_Interf.wav	20
	20__18_07_13_AutopridePrepManiobra.wav	20	69__23_07_13_H2_costaVoyager.wav	20
Background noise	82__27_09_13_H3_lluvia.wav	40	81__25_09_13_H3_corriente.wav	40
	86__E__2M.wav	40	84__27_09_13_H3_viento.wav	40

	3rd SUBSET	
	Recording	Time (s)
Dolphin	Dolphin2.wav	6.9
	Dolphin4.wav	7.9
	Dolphin3.wav	9.4
Humpback whale	Whale5.wav	17
	Whale8.wav	29.5
Small vessels	75__23_07_13_H3_pesquero1.wav	20
	26__19_07_13_Lancha.wav	20
Large vessels	13__10_07_13_piraCies_Entra.wav	20
	24__19_07_13_adventureFrenando_duda.wav	20
Background noise	88__E__4L.wav	40
	92__E__8H_N.wav	40

Table 5.8: Testing data subsets

Each model was trained for up to 50 epochs and with a batch size of 200. The model parameters of epoch 50 were the ones selected to be evaluated in the testing data, since training loss and accuracy suffered minor changes after epoch 40. There was a little uncertainty in the performance of the model in each evaluation of the three testing subsets due to the small size of the training data set. This led to some differences in the performance estimation from one run of the three testing subsets to another one. In order to reduce this uncertainty and to have a more reliable estimation, the model performance was evaluated using five repeats of the three testing subsets [38], generating 15 different estimates of the accuracy.

5.2.5 Data augmentation results

Before analysing the performance achieved by the five models with different window length, the impact of the three data augmentation techniques (time stretching, pitch shifting and time shifting) was studied. In this way, using the current recognition framework, the three augmentation techniques were applied individually and simultaneously to training data of dolphin and humpback whale classes. Thus, four models were built. An additional model was also assembled without data augmentation. Augmentation techniques were only applied to training data, in order to avoid biased results in testing data. The five models were constructed using the same window length, **1.00 s** (50 frames).

The results of the five models are reported in Figure 5.6 as a box-and-whisker plot generated from the 15 different estimates of the accuracy for each model. The cross and horizontal line represent the average and median accuracy, respectively. The three augmentation techniques applied individually achieved an average accuracy equal to 73.7%, 76.3%, 75.5%, corresponding to the application of time stretching, pitch shifting and time shifting, respectively. This corresponds to an improvement in comparison with the model without data augmentation, which achieved an average accuracy of 71.6%. The highest improvement was achieved by using the three augmentation techniques simultaneously, with a corresponding average accuracy of 76.5%.

In the following study about the impact of the window length, all models were constructed by applying the three augmentation techniques simultaneously to training data of dolphin and humpback whale classes.

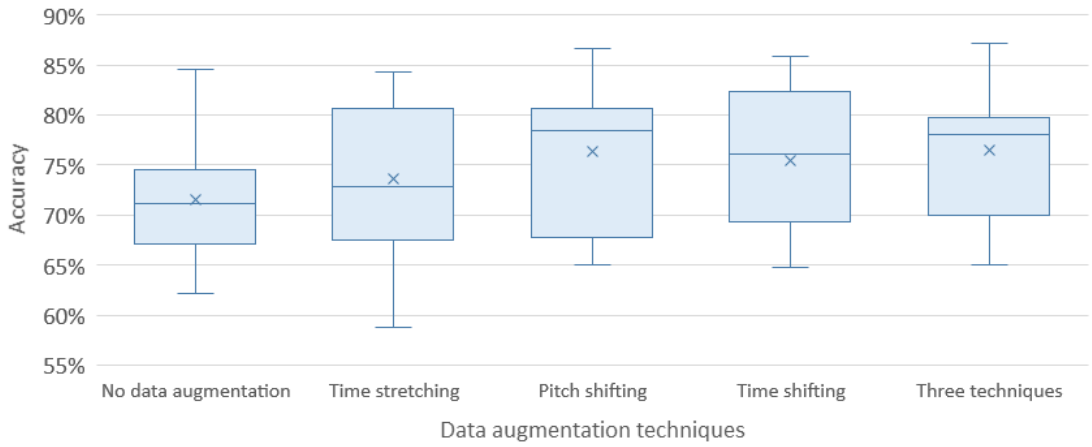

Figure 5.6: Repeated accuracy results using different data augmentation techniques

5.2.6 Results of window length variation

A first overview of the performance of each model is based, once again, on the 15 different estimates of the accuracy, which are represented as a box-and-whisker plot in Figure 5.7. The cross and horizontal line represent the average and median accuracy, respectively. The average accuracy of each model was 66.2%, 69.4%, 76.5%, 78.3% and 72.1%, corresponding to the window length of 0.22 s, 0.61 s, 1.00 s, 1.48 s and 1.97 s, respectively. There was an increase in the average accuracy up to window length of 1.48 s. An improvement was already expected when going from extremely small windows (0.22 s and 0.61 s) to intermediate windows (1.00 s or 1.48 s), since these contain more information allowing more accurate predictions. Nevertheless, the accuracy rate decreased when using a window length of 1.97 s.

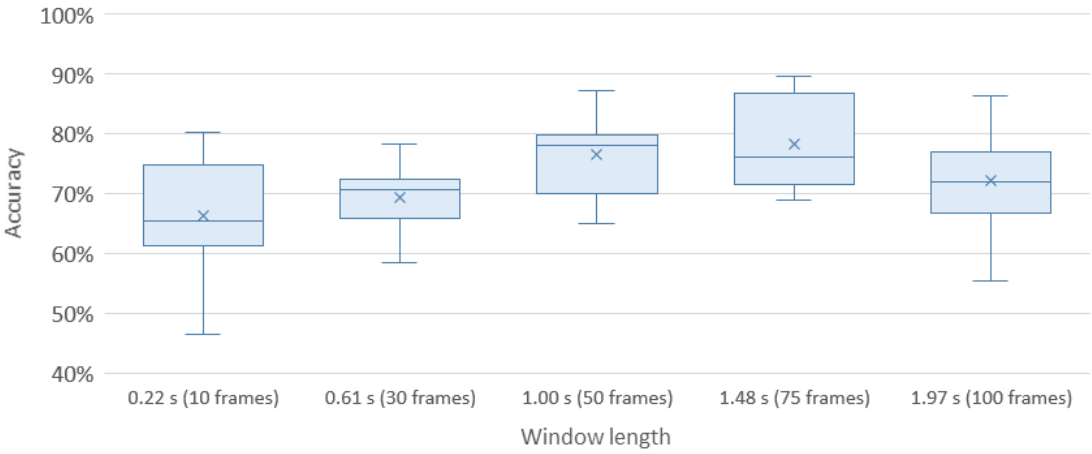

Figure 5.7: Repeated accuracy results using different window length

Tables 5.9-5.11 and the complementary confusion matrix of each model in Appendix C allow a more detailed analysis based on precision, recall and F1-score metrics (see Equations 18-20) for each class.

The confusion matrices correspond to the best repeat of the the three testing subsets of each model taking into account the accuracy rate. The precision, recall and F1-score were computed as the average of those metrics among the five repeats.

	Dolphins	Humpback Whales	Small Vessels	Large Vessels	Background Noise
window length: 0.22 s (10 frames)	78.1	91.5	51.0	47.9	76.6
window length: 0.61 s (30 frames)	80.6	91.8	61.7	47.3	75.4
window length: 1.00 s (50 frames)	78.0	92.0	68.5	63.1	78.6
window length: 1.48 s (75 frames)	82.4	88.8	68.2	65.1	83.3
window length: 1.97 s (100 frames)	70.3	84.7	59.0	58.5	86.5

Table 5.9: Classifier precision (%)

	Dolphins	Humpback Whales	Small Vessels	Large Vessels	Background Noise
window length: 0.22 s (10 frames)	70.5	56.3	60.6	60.6	76.1
window length: 0.61 s (30 frames)	60.0	64.8	66.2	66.1	80.6
window length: 1.00 s (50 frames)	64.1	90.1	80.3	64.0	80.8
window length: 1.48 s (75 frames)	65.0	94.5	83.6	64.1	81.7
window length: 1.97 s (100 frames)	68.2	100	83.0	56.3	62.5

Table 5.10: Classifier recall (%)

	Dolphins	Humpback Whales	Small Vessels	Large Vessels	Background Noise
window length: 0.22 s (10 frames)	74.0	68.4	54.4	52.0	76.3
window length: 0.61 s (30 frames)	68.5	75.6	63.8	55.0	77.8
window length: 1.00 s (50 frames)	70.3	91.0	73.9	63.4	79.7
window length: 1.48 s (75 frames)	72.7	91.5	74.9	63.9	82.4
window length: 1.97 s (100 frames)	69.0	91.5	68.5	56.6	72.5

Table 5.11: Classifier F1-score (%)

As for the classifier accuracy, the conclusion is that the best performances were achieved by the two models with intermediate windows (1.00 s and 1.48 s), where the F1-scores for each class show a similar performance for both, but with a slightly improvement for the model with a window length of 1.48 s. These two models always achieved a F1-score above 70.0% for all classes, with the exception of large vessel class. This is quite positive since it indicates that an interval between 1.00 s and 1.48 s is enough to make a reliable prediction.

In the case of the dolphin class, there was a similar performance for the five models, where the F1-scores ranged between 68.5% and 74.0%. The recall results shows that were a few misclassifications of dolphin data, which may be confirmed in the confusion matrix (see Appendix C) of each model. This indicates that

some dolphin windows were wrongly classified as background noise. The same misclassification occurred for a few background noise windows, albeit to a lesser degree, which were classified as dolphins. This mistake was due to the fact that two characteristic vocalizations produced by dolphins were added in the same class, which are the whistles and the echolocation clicks. These two sounds are very different leading to a great difference in the mel spectrogram representation. As explained in the description of the dataset, whistles give rise to high frequency contours and echolocation clicks create vertical lines. The vertical lines are not very pronounced in the mel spectrogram, which may lead them to be confused with the background noise, which is not represented by any particular frequency. In addition, a few mel spectrogram representations of background noise also contain vertical lines due to particular sources, such as the sound of the hydrophone hitting the water or even the sound of the rain.

There was a sharp improvement in the case of the recognition of humpback whale vocalizations with the increase of the window length. The F1-score increased from 68.4% to 91.5% and this behaviour is even more emphasized in the recall results, which increased from 56.3% to 100% . In contrast to the precision for the humpback whale class, which was always above 84.7% with the increase of the window length, the recall results shows a dramatic improvement. In order to understand the positive impact of a larger window dimension in the performance regarding this class, we should look at the confusion matrix of the model with a window duration of 0.22 s (see Appendix C). As may be seen, 33% of the humpback whale windows were wrongly classified as small or large vessels. This mistake might be due to both humpback whale vocalizations and vessel noise having a higher energy at low frequencies. However, if the window duration is increased, it may be perceived that whale vocalizations evolve over time, being in that case then represented as an arc in the mel spectrogram. This becomes a distinguishing factor between the two classes, since, in contrast to whale vocalizations, vessel noise is constant.

The classes with the lowest precision were the small and large vessels, which may be explained by the confusion between humpback whale vocalizations and vessel noise for the models with smaller windows as explained above, but also because of a few misclassifications between both vessel classes. This last reason also explains the lower recall for the large vessel class, and also for the small vessel class of the two models with smaller windows. Comparing the results with the two models constructed only with data from the ShipsEar dataset (Chapter 5.1), one can conclude that there was a decrease in the performance for the vessel classes, which was already expected since, in this case, data splitting was performed by recording. This leads to a more challenging problem.

Table 5.12 shows the vessel recall, which considers as true positives windows correctly classified as vessel noise, regardless whether these are from small or large vessels. The vessel recall was always above 93.7%, showing that rarely were any vessel windows classified as either animal vocalizations or background noise for the five models, which was quite positive.

	Vessel recall
window length: 0.22 s (10 frames)	93.7
window length: 0.61 s (30 frames)	96.4
window length: 1.00 s (50 frames)	97.2
window length: 1.48 s (75 frames)	97.6
window length: 1.97 s (100 frames)	95.2

Table 5.12: Classifier recall (%) of the two vessel classes

All models, except the one with a window duration of 1.97 s, correctly identified the absence of human activity or marine animals in most of the windows, where the background noise recall ranged between 76.1% and 81.7%.

5.2.7 Graphical representation of the results

To conclude the topic results, a few recordings are analysed individually based on the class scores. This allows a better understanding of all the comments made in the discussion of the results. In addition, in this way, one can have a better comprehension of the class scores assigned to each window and thus visualize how this methodology can be used in a real-time classification.

The following diagrams correspond to the signal waveform attached to the class scores for each window. These were predicted by the model over time and were represented as bars. A different colour was assigned to each class, as seen in each diagram. Intermittent lines were used to define the end of each window. Time is represented by three decimal places, unlike the two decimal places that were employed elsewhere. This was important in terms of the conversion of window length from frames to seconds, in order to be more accurate about the exact time of the end of each window.

The first two examples (Figures 5.8 and 5.9) correspond to the classification performed by the model using a window length of 1.48 s to the *DolphinWavEC1.wav* and *DolphinWavEC2.wav* recordings. Both contain the two characteristic dolphin vocalizations (whistles and echolocation clicks). In *DolphinWavEC1.wav*, all windows were correctly classified with very high confidence values. In *DolphinWavEC2.wav*, there is a small group of windows classified correctly with high confidence values. However, most of the windows were classified as background noise which matches the confusion between dolphin and background noise, already described. Most of the misclassification occurred in the echolocation clicks.

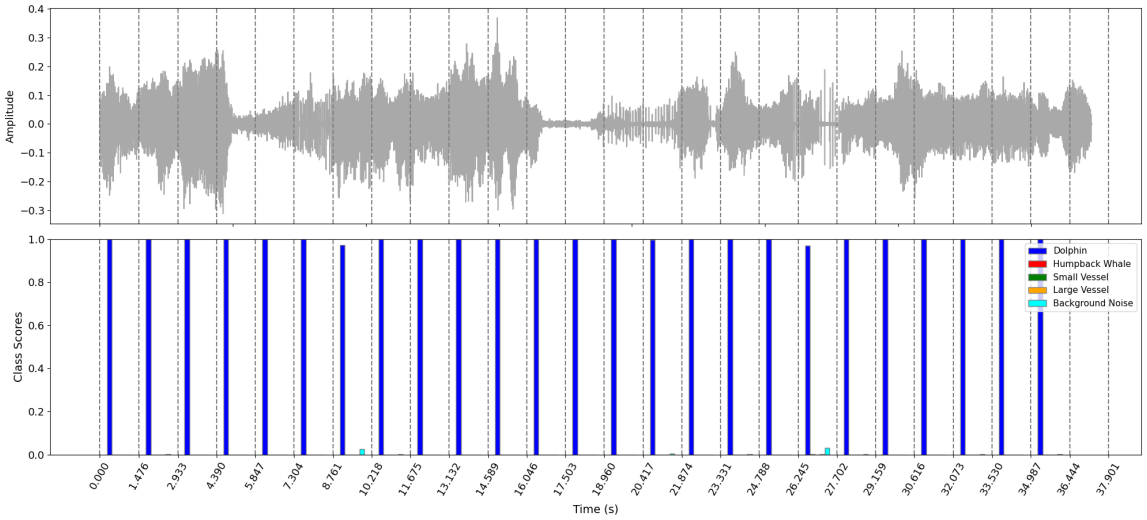

Figure 5.8: Signal waveform and classification (window length: 1.48 s) - Dolphin (1)
DolphinWavEC1.wav

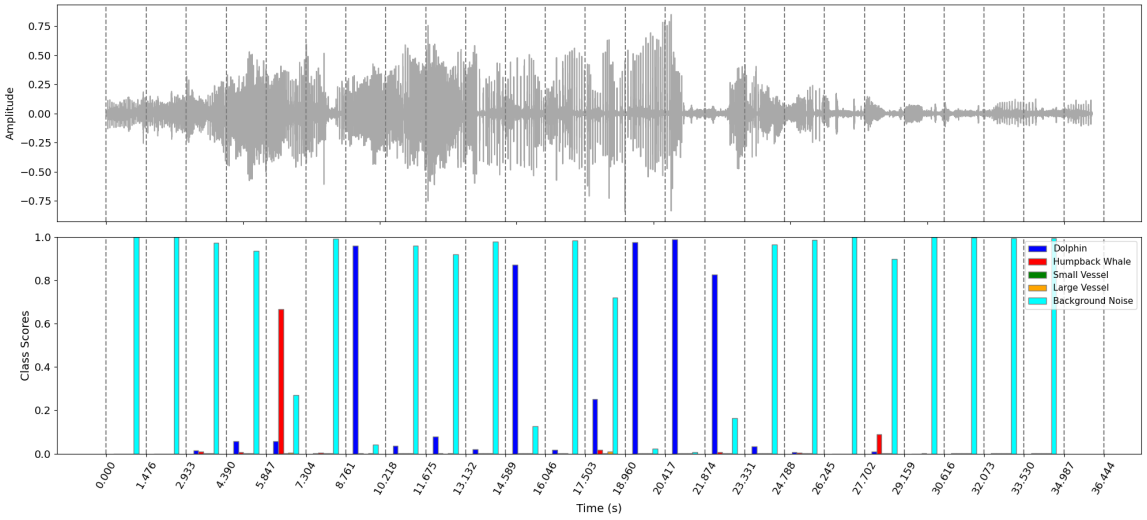

Figure 5.9: Signal waveform and classification (window length: 1.48 s) - Dolphin (2)
DolphinWavEC2.wav

The next two examples (Figures 5.10 and 5.11) correspond to the same recording (*Whale6.wav*), however classified with different models. The first graph corresponds to the class scores predicted by the model with a window length of 0.61 s and the second one with 1.48 s. The wider window length helps the model to detect the variations of amplitude in whale vocalizations, leading, in this case, to a drastic improvement in the classification of the humpback whale when changing a window length from 0.61 s to 1.48 s.

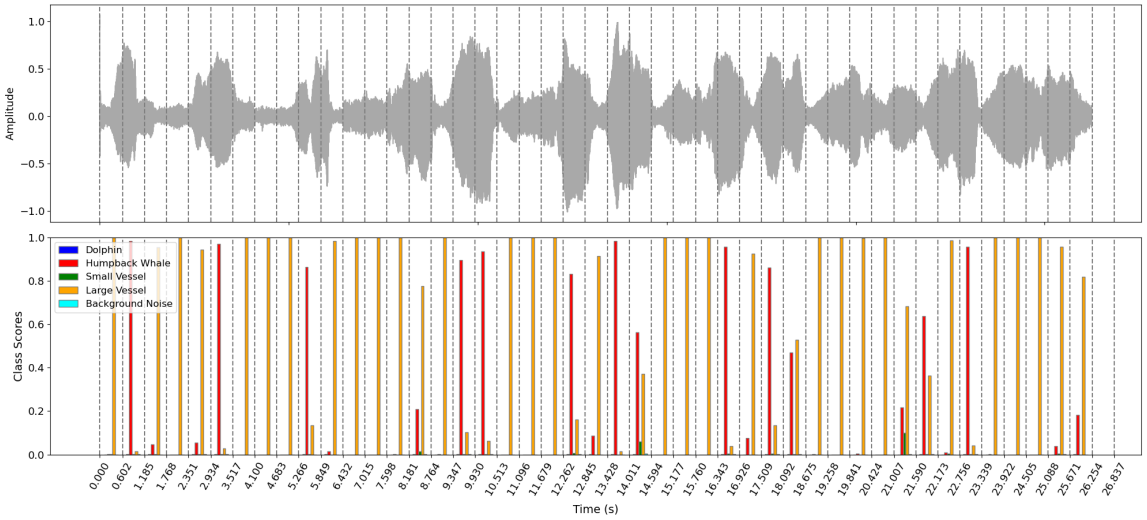

Figure 5.10: Signal waveform and classification (window length: 0.61 s) - Humpback whale *Whale6.wav*

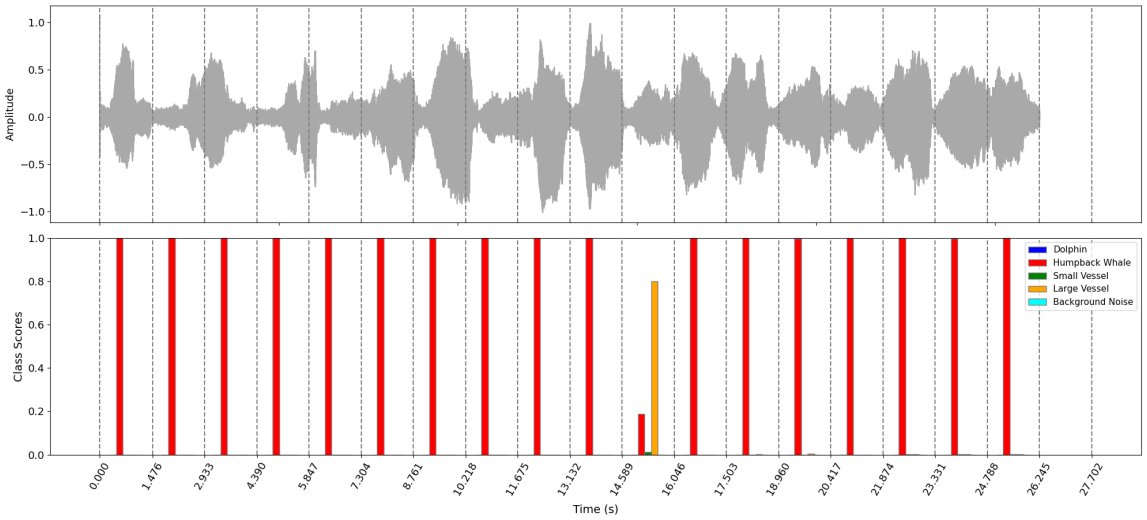

Figure 5.11: Signal waveform and classification (window length: 1.48 s) - Humpback whale *Whale6.wav*

The last example (Figures 5.12) corresponds to a recording of a large vessel classified by the model with a window length of 1.48s. Although some windows were incorrectly classified as small vessels, the model leaves no doubts about the presence of vessel noise in the recording, which shows that it is quite reliable for vessel detection.

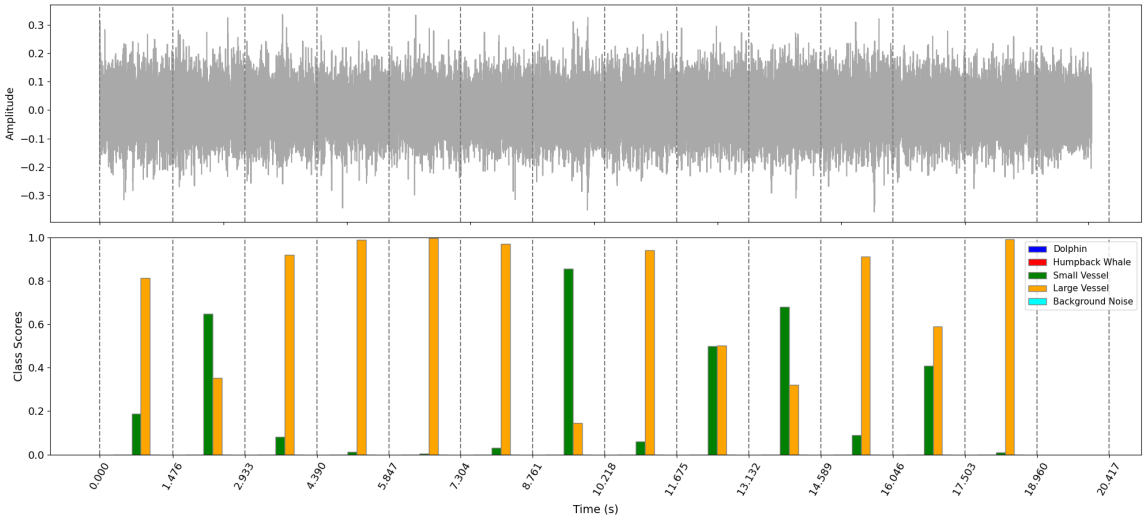

Figure 5.12: Signal waveform and classification (window length: 1.48 s) - Large vessel *24__19_07_13_adventureFrenando_duda.wav*

Chapter 6

Conclusions

This thesis was based on a project proposed by the WavEC company, which consists in developing a model that is capable of recognizing sea sound sources (such as marine animals, man-made activity or natural sea sounds) that are present in hydroacoustic "raw" data. The task was conducted as a ML problem, where a series of hydroacoustic signals were used to find common patterns in the data. The methodology behind the model consisted in converting the "raw" data into more representative features by applying signal processing techniques. These features were classified by a state-of-the-art ML: CNN.

Two different feature extractions were studied, based on previous works in the field of ASR. Both involved converting the acoustic signal to a mel spectrogram representation. However, an extra step was added to one of the feature extraction processes, which consisted in computing the first and second derivatives of the mel spectrogram.

After introducing the recognition framework, this was applied to three different datasets. The first corresponds to the UrbanSound8k dataset, which was used solely as a testing of the framework. This step was important, since, albeit the data was not composed by underwater acoustic signals, but rather of urban sound acoustic data, it allowed to understand the capabilities of the methodology with a large dataset and compare the results with those of other works using that same dataset. In this case, only the mel spectrogram was used as input of the CNN, and an analysis of the impact of the window length on the classification accuracy was performed. Five different models were constructed with different window length. The average accuracy of the five models ranged between 69.3% and 72.2%, which are very close to the level of other works using a similar methodology.

ShipsEar was the second dataset used in this work. ShipsEar consists of underwater acoustic data of various types of vessels and background noise. Its application had an objective to study the performance of the model in identifying the presence and absence of vessel noise and distinguishing the vessel noise into four different vessel classes, based on the vessel size. This dataset was provided by Santos-Domínguez et al. [33]. The two feature extractions were applied with a fixed window length of 1.00 s (50 frames). Based on accuracy, precision, recall and F1-score, one can conclude that the model using the feature combination between the mel spectrogram and the first and second derivatives performed better than only using the

mel spectrogram as features, with a corresponding average accuracy of 88.8% and 83.2%, respectively. In addition, the results showed a good performance for both models in the perception of the presence and absence of vessel noise. However, there were a few misclassifications when assigning the vessel noise into the correct vessel class.

The last dataset aimed to spread the spectrum of marine sound sources. Thus, the data from the ShipsEar dataset was complemented with a set of marine animal (dolphin and humpback whale) recordings, which were either acquired from open sources or provided by WavEC. In this way, the recognition methodology was evaluated in a wider scenario, which was composed by vessel noise, natural sea sounds and marine animal vocalizations. Only the combination of the mel spectrogram and its first and second derivatives was used as features. Each model was evaluated using five repeats of the three testing subsets. Three data augmentation techniques (time stretching, pitch shifting and time shifting) were studied. These were applied individually and simultaneously to training data of dolphin and humpback whale classes. Based on the average accuracy of the five repeats of the three testing subsets, the model with the three techniques applied simultaneously outperformed the model without data augmentation, with a corresponding accuracy rate of 76.5%. Once again, the influence of the window length in the classification accuracy was analysed. The results indicated that the best two models used a window length of 1.00 s and 1.48 s, which achieved an accuracy rate of 76.5% and 78.3%, respectively. In both cases, the model correctly recognized the majority of the humpback whale vocalizations with a recall above 90.1%. Regarding the dolphin class, the recall of around 65% showed that there were a few dolphin windows misclassified as background noise. There was a good performance in detecting the presence of vessel noise, however with a few wrong classifications between the two vessel classes. Moreover, the model showed a reliable performance in recognizing the presence of purely natural sea sounds. In this way, the devised methodology achieved the intended objective, which indicated that it could be a reliable tool in a future real-time application, since the two best models make predictions for windows with less than 1.50 s.

This work may be a springboard for further and more detailed research in this area. New challenges can be tackled using a higher number of recordings for all classes, in order to study the capacities of the recognition framework in more detail. A review of the labeling process could be made by individualizing aspects that characterize each class. In order to avoid some classification mistakes for the dolphin data, partitioning this class based on the types of vocalizations (whistles and echolocation clicks) would be an improvement. The same should take place with the background noise class by distinguishing the recordings in which the presence of atmosphere elements (such as rain or wind) is more evident. In this field, other works also used this strategy of bracketing similar sounds from the same source [14]. Moreover, for vessels, performing an analysis based on the type of vessel rather than its size might be more suitable.

Although the positive results achieved by the model using the conversion of the acoustic signal into a mel spectrogram representation as baseline feature extraction process, there are other processes that can

be explored in order to add more representative features to the input of the CNN, such as studying the application of MFCCs.

The designed methodology cannot distinguish and correctly classify overlapping sounds. Therefore, this could be an interesting area to explore in order to devise a model which is better prepared to address the different scenarios which may occur in a marine environment.

References

[1] Y. Xu, Y. Zhang, H. Wang, and X. Liu, "Underwater image classification using deep convolutional neural networks and data augmentation," *2017 IEEE International Conference on Signal Processing, Communications and Computing, ICSPCC 2017*, vol. 2017-January, pp. 1–5, 2017. DOI: 10.1109/ICSPCC.2017.8242527.

[2] S. Marini, E. Fanelli, V. Sbragaglia, E. Azzurro, J. Del Rio Fernandez, and J. Aguzzi, "Tracking fish abundance by underwater image recognition," *Scientific Reports*, vol. 8, no. 1, pp. 1–12, 2018, ISSN: 20452322. DOI: 10.1038/s41598-018-32089-8.

[3] F. Han, J. Yao, H. Zhu, and C. Wang, "Underwater image processing and object detection based on deep cnn method," *Journal of Sensors*, vol. 2020, 2020, ISSN: 16877268. DOI: 10.1155/2020/6707328.

[4] N. Wawrzyniak, T. Hyla, and A. Popik, "Vessel detection and tracking method based on video surveillance," *Sensors (Switzerland)*, vol. 19, no. 23, 2019, ISSN: 14248220. DOI: 10.3390/s19235230.

[5] I. Jeon, S. Ham, J. Cheon, A. M. Klimkowska, H. Kim, K. Choi, and I. Lee, "A real-time drone mapping platform for marine surveillance," *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences - ISPRS Archives*, vol. 42, no. 2/W13, pp. 385–391, 2019, ISSN: 16821750. DOI: 10.5194/isprs-archives-XLII-2-W13-385-2019.

[6] J. Huang, J. Zhao, and Y. Xie, "Source classification using pole method of ar model," in *1997 IEEE International Conference on Acoustics, Speech, and Signal Processing*, IEEE, vol. 1, 1997, pp. 567–570.

[7] R. C. Bennett *et al.*, "Classification of underwater signals using a back-propagation neural network," PhD thesis, Naval Postgraduate School, 1997.

[8] A. Kundu, G. C. Chen, and C. E. Persons, "Transient sonar signal classification using hidden markov models and neural nets," *IEEE Journal of Oceanic Engineering*, vol. 19, no. 1, pp. 87–99, 1994, ISSN: 15581691. DOI: 10.1109/48.289454.

[9] B. Howell and S. Wood, "Passive sonar recognition and analysis using hybrid neural networks," in *Oceans 2003. Celebrating the Past... Teaming Toward the Future (IEEE Cat. No. 03CH37492)*, IEEE, vol. 4, 2003, pp. 1917–1924.

[10] C. Kang, X. Zhang, A. Zhang, and H. Lin, “Underwater acoustic targets classification using welch spectrum estimation and neural networks,” in *International Symposium on Neural Networks*, Springer, 2004, pp. 930–935.

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[12] H. A. Garcia, T. Couture, A. Galor, J. M. Topple, W. Huang, D. Tiwari, and P. Ratilal, “Comparing performances of five distinct automatic classifiers for fin whale vocalizations in beamformed spectrograms of coherent hydrophone array,” *Remote Sensing*, vol. 12, no. 2, pp. 1–25, 2020, ISSN: 20724292. DOI: 10.3390/rs12020326.

[13] M. Harvey *et al.*, “Acoustic detection of humpback whales using a convolutional neural network,” *Google AI Blog*, 2018.

[14] E. H. Belghith, F. Rioult, and M. Bouzidi, “Acoustic diversity classifier for automated marine big data analysis,” *Proceedings - International Conference on Tools with Artificial Intelligence, ICTAI*, vol. 2018-Novem, pp. 130–136, 2018, ISSN: 10823409. DOI: 10.1109/ICTAI.2018.00029.

[15] J. Salamon, C. Jacoby, and J. P. Bello, “A dataset and taxonomy for urban sound research,” *MM 2014 - Proceedings of the 2014 ACM Conference on Multimedia*, no. 1, pp. 1041–1044, 2014. DOI: 10.1145/2647868.2655045.

[16] J. Nordby, “Environmental sound classification on microcontrollers using convolutional neural networks,” Master’s thesis, Norwegian University of Life Sciences, May 2019. [Online]. Available: http://hdl.handle.net/11250/2611624.

[17] J. Salamon and J. P. Bello, “Deep convolutional neural networks and data augmentation for environmental sound classification,” *IEEE Signal Processing Letters*, vol. 24, no. 3, pp. 279–283, 2017.

[18] B. Mehlig, “Machine learning with neural networks,” 2021. arXiv: arXiv:1901.05639v3.

[19] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, “From data mining to knowledge discovery in databases,” *AI magazine*, vol. 17, no. 3, pp. 37–37, 1996.

[20] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors,” *Nature*, vol. 323, no. 6088, pp. 533–536, 1986, ISSN: 00280836. DOI: 10.1038/323533a0.

[21] *Cs231n: Convolutional neural networks for visual recognition*, https://cs231n.github.io/convolutional-networks/, Accessed: 2021-05-30.

[22] *Cs231n: Convolutional neural networks for visual recognition*, https://cs231n.github.io/linear-classify/, Accessed: 2021-05-30.

[23] *Cs231n: Convolutional neural networks for visual recognition*, https://cs231n.github.io/optimization-1/, Accessed: 2021-05-30.

[24] NOAA, *What is a thermocline?* https://oceanservice.noaa.gov/facts/thermocline.html, 25/02/21.

[25] ——, *How far does sound travel in the ocean?* https://oceanservice.noaa.gov/facts/sound.html, 25/02/21.

[26] ——, *What is sofar?* https://oceanservice.noaa.gov/facts/sofar.html, 25/02/21.

[27] ——, *What is sonar?* https://oceanservice.noaa.gov/facts/sonar.html, 25/02/21.

[28] M. H. J. Gruber and M. H. Hayes, *Statistical digital signal processing and modeling*, 1997. DOI: 10.2307/1271141.

[29] P. Dimitris, "Digital signal processing: Principles, devices and applications," *Digital Signal Processing: principles, devices and applications*, 1990. DOI: 10.1049/pbce042e.

[30] A. Georgescu, "Introduction to digital spectral analysis techniques," *Transactions on Mechanics, Scientific Bulletin of the "Politehnica" University of Timişoara*, vol. 50, p. 64, 2005.

[31] J. O. Smith III, *Spectral audio signal processing*. W3K publishing, 2011.

[32] H. Fayek, "Speech processing for machine learning: Filter banks, mel-frequency cepstral coefficients (mfccs) and what's in-between," *URL: https://haythamfayek. com/2016/04/21/speech-processingformachine-learning. html*, 2016.

[33] D. Santos-Domínguez, S. Torres-Guijarro, A. Cardenal-López, and A. Pena-Gimenez, "Shipsear: An underwater vessel noise database," *Applied Acoustics*, vol. 113, pp. 64–69, 2016, ISSN: 1872910X. DOI: 10.1016/j.apacoust.2016.06.008.

[34] I. Guyon, S. Gunn, M. Nikravesh, and L. A. Zadeh, *Feature extraction: foundations and applications*. Springer, 2008, vol. 207.

[35] B. McFee, V. Lostanlen, A. Metsai, M. McVicar, S. Balke, C. Thomé, C. Raffel, F. Zalkow, A. Malek, Dana, K. Lee, O. Nieto, J. Mason, D. Ellis, E. Battenberg, S. Seyfarth, R. Yamamoto, K. Choi, viktorandreevichmorozov, J. Moore, R. Bittner, S. Hidaka, Z. Wei, nullmightybofo, D. Hereñú, F.-R. Stöter, P. Friesch, A. Weiss, M. Vollrath, and T. Kim, *Librosa/librosa: 0.8.0*, version 0.8.0, Jul. 2020. DOI: 10.5281/zenodo.3955228. [Online]. Available: https://doi.org/10.5281/zenodo.3955228.

[36] P. Peso Parada and A. Cardenal-López, "Using gaussian mixture models to detect and classify dolphin whistles and pulses," *The Journal of the Acoustical Society of America*, vol. 135, no. 6, pp. 3371–3380, 2014, ISSN: 0001-4966. DOI: 10.1121/1.4876439.

[37] M. Grandini, E. Bagli, and G. Visani, “Metrics for multi-class classification: An overview,” *arXiv*, pp. 1–17, 2020, ISSN: 23318422. arXiv: 2008.05756.

[38] M. Kuhn and K. Johnson, *Applied predictive modeling*. 2013, vol. 26, p. 615, ISBN: 9781461468486. [Online]. Available: http://appliedpredictivemodeling.com/s/Applied%7B%5C_%7DPredictive%7B%5C_%7DModeling%7B%5C_%7Din%7B%5C_%7DR.pdf.

[39] A. Géron, *Hands on machine learning with scikit learn keras and tensorFlow 2nd edition-2019*, 9. 2019, vol. 53, p. 287, ISBN: 9788578110796. arXiv: arXiv:1011.1669v3.

[40] C. Li, Z. Liu, J. Ren, W. Wang, and J. Xu, “A feature optimization approach based on inter-class and intra-class distance for ship type classification,” *Sensors (Switzerland)*, vol. 20, no. 18, pp. 1–12, 2020, ISSN: 14248220. DOI: 10.3390/s20185429.

[41] X. Ke, F. Yuan, and E. Cheng, “Integrated optimization of underwater acoustic ship-radiated noise recognition based on two-dimensional feature fusion,” *Applied Acoustics*, vol. 159, p. 107 057, 2020, ISSN: 1872910X. DOI: 10.1016/j.apacoust.2019.107057. [Online]. Available: https://doi.org/10.1016/j.apacoust.2019.107057.

[42] R. Wu, S. Yan, Y. Shan, Q. Dang, and G. Sun, “Deep image: Scaling up image recognition,” *arXiv preprint arXiv:1501.02876*, vol. 7, no. 8, 2015.

[43] G. Maguolo, M. Paci, L. Nanni, and L. Bonan, “Audiogmenter: A matlab toolbox for audio data augmentation,” *arXiv*, 2019, ISSN: 23318422.

[44] S. Wei, S. Zou, F. Liao, and W. Lang, “A comparison on data augmentation methods based on deep learning for audio classification,” *Journal of Physics: Conference Series*, vol. 1453, no. 1, 2020, ISSN: 17426596. DOI: 10.1088/1742-6596/1453/1/012085.

Appendix A | ShipsEar dataset base information

Mentioned in Chapters 5.1 and 5.2

Recording	Class	Interval (s)
6__10_07_13_marDeCangas_Entra.wav	C	[120:160]
7__10_07_13_marDeCangas_Espera.wav	C	-
8__10_07_13_marDeOnza_Entra.wav	C	[15:55]
9__10_07_13_marDeOnza_Espera.wav	C	-
10__10_07_13_marDeOnza_Sale.wav	C	[0:40]
11__10_07_13_minhoUno_Entra.wav	C	[15:55]
12__10_07_13_minhoUno_Sale.wav	C	[30:70]
13__10_07_13_piraCies_Entra.wav	C	[120:160]
14__10_07_13_piraCies_Espera.wav	C	-
15__10_07_13_radaUno_Pasa.wav	A	[35:60]
16__10_07_13_mscOpera_InicioSalida.wav	D	[30:60]
17__10_07_13_visionSub_Entra.wav	C	[30:70]
18__18_07_13_AutoPrideEntra.wav	D	[120:240]
19__18_07_13_AutoprideMarchaAtras.wav	D	[120:240]
20__18_07_13_AutopridePrepManiobra.wav	D	Entire recording
21__18_07_13_lanchaMotora.wav	B	[6:25]
22__19_07_13_adventure_maniobra.wav	D	Entire recording
23__19_07_13_adventure_parado.wav	D	-
24__19_07_13_adventureFrenando_duda.wav	D	Entire recording
25__19_07_13_adventureOfTheSea_llegando.wav	D	[0:120]
26__19_07_13_Lancha.wav	B	Entire recording
27__19_07_13_Lancha2.wav	B	[8:14]
28__19_07_13_NuevoRiaAldan.wav	A	[0:60]
29__19_07_13_practico.wav	B	[0:60]
30__19_07_13_practico2.wav	B	[0:40]
31__19_07_13_RemolcadorArrancaPara.wav	A	[0:60]
32__19_07_13_Arroios_llega.wav	C	[120:170]
33__19_07_13_lanchaDuda_Sale.wav	B	[9:22]
34__19_07_13_MarDeOnza_Llega.wav	C	[0:40]

35__19_07_13_MarDeOnza_sale.wav	C	[0:40]
36__19_07_13_MinoUno_Sale.wav	C	[40:80]
37__19_07_13_Velero_Sale.wav	B	-
38__19_07_13_Arroios_Sale_2entran.wav	C	[30:70]
39__19_07_13_lanchaMotora_Entra.wav	B	[0:60]
40__19_07_13_MarDeCangas_Llega_Interf.wav	C	Entire recording
41__19_07_13_MinoUno_Entra.wav	C	[0:40]
42__19_07_13_PirataCies_Sale.wav	C	[0:40]
43__19_07_13_PirataDeCies_Llega_Interf.wav	C	[0:40]
45__19_07_13_yate_Sale.wav	B	[0:75]
46__23_07_13_H3_bateero1.wav	A	Entire recording
47__23_07_13_H2_bateero2.wav	A	[60:180]
48__23_07_13_H3_bateero3_interf.wav	A	Entire recording
49__23_07_13_H1_bateero4.wav	A	[60:120]
50__23_07_13_H3_lancha1.wav	B	Entire recording
51__23_07_13_H3_lancha2_interf.wav	B	Entire recording
52__23_07_13_H3_lancha3_interf.wav	B	Entire recording
53__23_07_13_H3_PirataSalvora_Llega1.wav	C	[0:50]
54__23_07_13_H3_PirataSalvora_SALE1.wav	C	[0:50]
55__23_07_13_H3_PirataSalvora_SALE2.wav	C	[0:50]
56__23_07_13_H3_velero1.wav	B	Entire recording
57__23_07_13_H3_velero2.wav	B	Entire recording
58__23_07_13_H2_EimskipReefer_Pesquerito1.wav	D	[60:180]
59__23_07_13_H2_MarDeMouro.wav	C	[90:140]
60__23_07_13_H3_MarDeCangas_LLEGA.wav	C	[120:170]
61__23_07_13_H3_MarDeCangas_SALE-2.wav	C	[60:110]
62__23_07_13_H3_MarDeCangas_SALE.wav	C	[120:170]
63__23_07_13_H3_MarDeOnza_LLEGA.wav	C	[0:50]
64__23_07_13_H3_MarDeOnza_SALE.wav	C	[0:50]
65__23_07_13_H3_MinhoUno_LLEGA.wav	C	[0:50]
66__23_07_13_H3_Pesquerito2_Velero.wav	A	[0:120]
67__23_07_13_H3_PirataCiesSale.wav	C	[165:215]
68__23_07_13_H3_veleroMarisol.wav	B	[0:120]
69__23_07_13_H2_costaVoyager.wav	D	[60:180]

70__23_07_13_H2_yatePeq.wav	B	Entire recording
71__23_07_13_H3_Discovery_UK_DUDA.wav	D	[0:120]
72__23_07_13_H3_lancha2.wav	B	[0:120]
73__23_07_13_H3_pesqMariCarmen.wav	A	[0:120]
74__23_07_13_H3_pesqSaladinoPrimero.wav	A	[0:120]
75__23_07_13_H3_pesquero1.wav	A	Entire recording
76__23_07_13_H3_pesquero2.wav	A	[0:120]
77__23_07_13_H3_Planeadora.wav	B	Entire recording
78__23_07_13_H3_vikingChance.wav	D	[0:120]
79__23_07_13_H3_zodiac.wav	B	Entire recording
80__04_10_12_adricristuy.wav	A	Entire recording
81__25_09_13_H3_corriente.wav	E	Entire recording
82__27_09_13_H3_lluvia.wav	E	Entire recording
83__27_09_13_H3_oleaje.wav	E	Entire recording
84__27_09_13_H3_viento.wav	E	Entire recording
85__E__1L.wav	E	Entire recording
86__E__2M.wav	E	Entire recording
87__E__3H.wav	E	Entire recording
88__E__4L.wav	E	Entire recording
89__E__5M.wav	E	Entire recording
90__E__6H.wav	E	Entire recording
91__E__7H_N.wav	E	Entire recording
92__E__8H_N.wav	E	Entire recording
93__A__Draga_1.wav	A	Entire recording
94__A__Draga_2.wav	A	Entire recording
95__A__Draga_3.wav	A	Entire recording
96__A__Draga_4.wav	A	Entire recording

Appendix B | Marine animals website information

Mentioned in Chapter 5.2

Name (Table 5.6)	Website	Website Name
Dolphin1.wav	http://www.lpi.tel.uva.es/~nacho/docencia/ing_ond_1/trabajos_04_05/io5/public_html/sonidos.htm	common dolphin.mp3
Dolphin2.wav	https://www.fisheries.noaa.gov/national/science-data/sounds-ocean	Short-beaked Common Dolphin
Dolphin3.wav	http://www-3.unipv.it/cibra/edu_dolphins_uk.html	T_truncatus_whistles_short.mp3
Dolphin4.wav	www.polmar.com/sons/dauphinbb.wav	-
Dolphin5.wav	http://www-3.unipv.it/cibra/edu_dolphins_uk.html	S_coerulecalba_whistles_short.mp3
Whale1.wav	http://www.oceanmammalinst.org/songs.html	Song 1 (WAV)
Whale2.wav	http://www.oceanmammalinst.org/songs.html	Song 2 (WAV)
Whale3.wav	http://www.oceanmammalinst.org/songs.html	Song 3 (WAV)
Whale4.wav	http://www.oceanmammalinst.org/songs.html	Song 4 (WAV)
Whale5.wav	Website not found	-
Whale6.wav	https://ai.googleblog.com/2018/10/acoustic-detection-of-humpback-whales.html	-
Whale7.wav	https://www.fisheries.noaa.gov/national/science-data/sounds-ocean	Humpback Whale
Whale8.wav	https://www.nps.gov/glba/learn/nature/soundclips.htm	Glacier Bay 60-second clip

Appendix C | Confusion matrix

Mentioned in Chapter 5.2

The correct predictions along the diagonal (recall values), and the wrong classifications elsewhere

window length: 0.22 s (10 frames)

		Predicted class				
		Dolphins	Humpback whales	Small vessels	Large vessels	Background noise
True class	Dolphins	72.9%	2.9%	0.9%	0.3%	23.1%
	Humpback whales	3.2%	59.8%	22.5%	10.5%	4.0%
	Small vessels	1.1%	0.2%	71.0%	27.1%	0.6%
	Large vessels	0%	0%	25.4%	68.4%	6.2%
	Background noise	12.1%	0%	1.5%	10.3%	76.1%

window length: 0.61 s (30 frames)

		Predicted class				
		Dolphins	Humpback whales	Small vessels	Large vessels	Background noise
True class	Dolphins	63.6%	0.4%	1.5%	0.4%	34.1%
	Humpback whales	1.4%	69.3%	1.4%	24.2%	3.7%
	Small vessels	2.9%	0%	78.4%	17.2%	1.5%
	Large vessels	0.5%	12.7%	21.5%	65.4%	0%
	Background noise	1.2%	0.2%	0%	5.1%	82.6%

window length: 1.00 s (50 frames)

		Predicted class				
		Dolphins	Humpback whales	Small vessels	Large vessels	Background noise
True class	Dolphins	67.5%	1.3%	0.6%	0.6%	29.9%
	Humpback whales	1.6%	92.9%	2.4%	1.6%	1.6%
	Small vessels	0.8%	0%	86.7%	12.5%	0%
	Large vessels	0.8%	2.5%	22.3%	74.4%	0%
	Background noise	9.3%	0.8%	0.8%	2.8%	86.2%

window length: 1.48 s (75 frames)

		Predicted class				
		Dolphins	Humpback whales	Small vessels	Large vessels	Background noise
True class	Dolphins	64.1%	1.0%	2.9%	0%	32.0%
	Humpback whales	1.2%	95.2%	0%	3.6%	0%
	Small vessels	1.3%	0%	78.2%	20.5%	0%
	Large vessels	0%	5.1%	22.8%	72.2%	0%
	Background noise	10.5%	1.2%	0%	2.5%	85.8%

window length: 1.97 s (100 frames)

		Predicted class				
		Dolphins	Humpback whales	Small vessels	Large vessels	Background noise
True class	Dolphins	72.4%	1.3%	11.8%	0%	14.5%
	Humpback whales	0%	100%	0%	0%	0%
	Small vessels	6.7%	0%	88.3%	0.5%	0%
	Large vessels	6.7%	11.7%	41.7%	40.0%	0%
	Background noise	16.7%	7.5%	3.3%	9.2%	63.3%