UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
ESCUELA DE POSTGRADO Y EDUCACIÓN CONTINUA

ADAPTATIVE AND AUTOMATIC MANATEE VOCALIZATION DETECTION USING TRANSFORMERS

TESIS PARA OPTAR AL GRADO DE MAGÍSTER EN
CIENCIA DE DATOS

STEFANO GIOVANNI SCHIAPPACASSE MALDINI

PROFESOR GUÍA:
FELIPE TOBAR HENRIQUEZ

MIEMBROS DE LA COMISIÓN:
TACO DE WOLFF
JORGE SILVA SANCHEZ

SANTIAGO DE CHILE
2024

RESUMEN DE LA TESIS PARA OPTAR
AL GRADO DE MAGÍSTER EN CIENCIA DE DATOS
POR: STEFANO GIOVANNI SCHIAPPACASSE MALDINI
FECHA: 2024
PROF. GUÍA: FELIPE TOBAR HENRIQUEZ

DETECCIÓN ADAPTATIVA Y AUTOMÁTICA DE VOCALIZACIONES DE MANATÍES UTILIZANDO TRANSFORMERS

En el campo de la conservación, es esencial contar con herramientas para estimar la población de diferentes especies. Particularmente para los manatíes, los métodos convencionales para estimar su población son bastante costosos y conllevan numerosos desafíos logísticos. En este contexto, y considerando que los manatíes frecuentemente producen vocalizaciones bajo el agua, el uso de grabaciones acústicas pasivas para contar manatíes y estimar el tamaño de su población se ha vuelto una opción cada vez más popular entre los expertos en conservación.

La metodología estándar consiste en implementar etapas de eliminación de ruido, detección y clasificación de manera independiente, donde las dos primeras son ajustadas por expertos, dejando sin opción de generalizar la solución para trabajar con audios grabados en diferentes entornos. Este trabajo aborda este problema y propone un enfoque novedoso que permite detectar llamadas de manatíes en grabaciones de audio y adapta la solución a diferentes fuentes de datos, posibilitando su implementación en varios ambientes habitados por esta especie, mediante la implementación de una solución de extremo a extremo donde las etapas de eliminación de ruido y clasificación se ajustan juntas bajo un marco de entrenamiento de aprendizaje profundo.

Los resultados obtenidos muestran que, aunque la solución de extremo a extremo obtiene un recall del 91 %, al observar la métrica de precisión, su desempeño es peor que el modelo de clasificación por sí solo, obteniendo una puntuación de precisión del 71 %. Este trabajo aún está en curso, por lo que aún queda trabajo por hacer para mejorar estos resultados.

Acknowledgements

I deeply appreciate my entire support network, especially my family and my life partner, thank you all for supporting me unconditionally. This work belongs to all of you, and I am confident that this is just the starting point; the best is yet to come.

Table of Content

Index of Tables

Index of Figures

Chapter 1

Introduction

In any species conservation program, it is essential to be able to estimate the number of individuals that exist in a given geographic location in order to carry out management. In the case of manatees, aerial and sonar studies have been carried out to make this estimate (Ackerman, 1995). However, these approaches suffer from detection limitations, logistical problems, and are relatively expensive. In the past decade the detection of marine mammal sounds through passive acoustic monitoring (PAM) has been widely used due to its low cost and non-invasive implementation in the deployed environment (Castro et al., 2015). Through PAM, it is possible to detect vocalizations of different marine species and thus estimate the number of individuals. For this purpose, it is necessary for these species to emit sounds that can be captured by underwater microphones, also known as hydrophones. The antillean manatee, *Trichechus manatus*, is characterized as a marine mammal that constantly emits sounds of various kinds (Phillips et al, 2003), enabling the use of PAM to manage its conservation through the processing of these underwater recorded sounds.

There are studies aiming to count how many different individuals are identified in an audio segment, using both classical signal processing techniques[1, 2] and deep learning approaches[3]. The implemented methodology in these studies is characterized by a **denoising** stage, where noise is removed from the audio, a **detection** stage that seeks to discard audio segments without vocalizations, a **classification** stage where the detected audio segments are classified, and finally, a **counting** stage where unsupervised techniques are used for segmentation and identification of different individuals. These studies have the particularity of processing audio with the need for the expert choice of multiple hyperparameters through experimentation, limiting the solution to working with data from the same source and not allowing generalization to audios recorded in other environments.

This study addresses this issue by proposing a novel modeling approach that allows adapting the solution to work with audio from different sources, where the hyperparameters used in the denoising and classification stages are learned through signal processing and deep learning techniques using an end-to-end architecture.

The implemented methodology is summarized in Figure 4.1. It consists of 3 main stages. The first stage involves the **denoising** step, where *power spectral floor denoising* technique presented by Tobar et al.[4] is applied. This technique uses Fourier analysis to remove noise from audio segments by attenuating the frequencies most present in a characteristic noise window, known as the power spectral floor window, and consequently enhance the frequencies that do not correspond to background noise. The second stage is the **generation of time-frequency representation of audios** where the denoised audio segments are transformed

to spectrograms to get a more expressive version of the audio waveforms where we can see the evolution of frequency components of the signals over time. Finally a **classification** stage is implemented, where the spectrograms representation of the audio segments are passed through an Audio Spectrogram Transformer model (AST) that generates predictions. The predictions are then evaluated by computing the loss in conjunction with the labels, and the error is propagated back to the first stage. Both stages are interconnected within a training framework, making the solution end-to-end and allowing the parameters of both the denoise and classification stages to be learned together and thus ensure that the solution is capable of adapting to audio that has been recorded in different environments.

To test this methodology, manatee audio recordings from *ZooParc de Beauval & Beauval Nature* in France, provided by the ECOSUR foundation in collaboration with C-MINDS under the project *A Machine Learning Approach to better understand and protect Marine Mammals in Latin America and the Caribbean: the case of the manatee* funded by Google under the AI for social good program, are being used. The dataset consists of 20 audio sessions, with an average duration of 10 minutes each, where less than 1 % of the time corresponds to manatee vocalizations. The denoising and classification stages were tested independently and also they were trained together to validate our proposal. Regarding the audio denoising stage, the results shows that the power spectral denoising successfully attenuates frequencies from background noise and enhances the frequencies of the vocalization and, on the other hand, in the classification stage, the implemented model successfully identifies, more than 80 % of all vocalizations with a precision over 80 %. When both models are trained together, the results drop regarding to the classification model by it self, demonstrating that capability of the AST to perform well without the need to used denoised audios as inputs. Even though this results do not validate our proposal, we strongly think that this is a consequence of using data recorded in a Zoo, where background noise might not be as relevant as for audios recordings in real hábitats of the manatee. Also, we conclude that noise can even help the model to discriminate better between positive a negative classes, as the AST model used, was pretrained on noisy data.

Chapter 2

Background

2.1. Manatee monitoring

2.1.1. Importance of monitoring manatee populations

Monitoring manatee populations is of paramount importance due to the manifold roles these marine mammals play in ecosystems and their status as vulnerable or endangered species[5]. The data generated from monitoring efforts are instrumental in assessing and refining conservation strategies aimed at safeguarding manatees from threats like boat strikes, habitat loss, and pollution[6]. Beyond their charismatic presence, manatees serve as critical indicators of ecosystem health, particularly in relation to seagrass beds and aquatic vegetation. As herbivores, their feeding habits influence the balance of underwater flora, establishing them as keystone species with a profound impact on biodiversity. Moreover, manatees' sensitivity to changes in water quality makes them invaluable sentinels for detecting environmental degradation caused by pollution, thus enabling timely interventions[7]. The monitoring of manatee populations also contributes to our understanding of the effects of climate change on these creatures and their habitats, providing essential data for adaptive measures. Scientific research into manatee behavior, migration patterns, and reproduction enhances our knowledge base, informing effective conservation management. By serving as ambassadors for marine conservation, manatees contribute to public awareness and education, fostering a sense of responsibility and promoting sustainable practices. Additionally, the population data derived from monitoring efforts play a pivotal role in shaping legal and policy decisions, influencing regulations and initiatives aimed at ensuring the long-term survival of these species. In essence, monitoring manatee populations is a multidimensional endeavor with far-reaching implications for ecological balance, biodiversity maintenance, and the broader well-being of aquatic environments.

Manatee population monitoring has undergone a long journey, evolving from sporadic observations to sophisticated, technology-driven methods aimed at understanding and safeguarding these marine mammals. Initial population surveys in the 1970s and 1980s relied on aerial and boat-based counts to provide baseline data on manatee numbers and distribution[8]. Subsequent decades witnessed the advent of tagging and tracking studies using satellite and radio telemetry, offering insights into manatee movements and habitat use[9]. Aerial surveys became a standard method for estimating populations, especially in Florida and the Caribbean. The 21st century ushered in a new era with the introduction of acoustic monitoring, utilizing hydrophones to capture and analyze manatee vocalizations. The integration of these

technologies represents a paradigm shift, enhancing the efficiency, accuracy, and real-time capabilities of manatee population assessments[1]. Collaborative conservation efforts involving government agencies, non-profit organizations, and research institutions underscore the importance of standardized monitoring methodologies and data-sharing practices. The ongoing evolution of manatee population monitoring reflects a commitment to informed conservation strategies and the long-term well-being of these vulnerable marine mammals.

2.1.1.1. Manatee population monitoring overview

Traditional methods for monitoring manatee populations, including aerial surveys, boat-based counts, and tagging studies, have played crucial roles in providing foundational data but are accompanied by distinct limitations. Aerial surveys, conducted via manned or unmanned aircraft, offer expansive coverage and a comprehensive view of manatee habitats. However, they are susceptible to weather conditions, with factors like cloud cover and rough seas impacting visibility[8]. The approach provides snapshot assessments, potentially overlooking variations in manatee distribution over time and submerged individuals, particularly in turbid or densely vegetated waters. Boat-based counts, involving visual observations from watercraft, offer proximity and individual identification capabilities. Yet, these surveys can be logistically challenging, requiring substantial resources and time, and are sensitive to disturbances that may alter manatee behavior during observations[10]. Tagging and tracking studies, leveraging satellite or radio telemetry, yield valuable insights into manatee movements and behavior. However, the invasive nature of tagging, potential risks to manatees, and the finite lifespan of tags limit the scalability and long-term tracking capabilities of this method. Moreover, traditional methods face challenges related to data accuracy, resolution, and the inability to capture nocturnal behaviors, as well as limitations in temporal coverage, providing periodic snapshots rather than continuous monitoring[3]. The integration of advanced technologies, such as acoustic monitoring and automated analysis, addresses these limitations, offering a more comprehensive, continuous, and non-invasive approach to manatee population monitoring, essential for refining conservation and management strategies.

The limitations inherent in traditional methods of manatee population monitoring underscore the critical need for more advanced and accurate counting methodologies. Aerial surveys and boat-based counts, while foundational, are subject to environmental variables and logistical challenges that compromise their reliability and precision. Weather conditions, such as cloud cover or rough seas, can impede visibility during aerial surveys, potentially leading to underestimations or incomplete assessments of manatee populations[8]. Boat-based counts, while offering a close-up perspective, are resource-intensive, logistically challenging, and sensitive to disturbances that may alter manatee behavior during observations. Moreover, these methods provide intermittent snapshots, failing to capture the dynamic and nocturnal aspects of manatee behavior, thus limiting the temporal resolution of population assessments[10]. Tagging and tracking studies, while informative, are invasive, pose potential risks to manatees, and have practical constraints regarding scalability and tracking duration. The imperative for more advanced and accurate counting methods is evident in the quest for continuous, non-invasive, and scalable approaches that can overcome the limitations of traditional techniques. Such advancements are vital for achieving a nuanced understanding of manatee populations, identifying temporal and spatial variations, and responding effectively to conservation challenges. The integration of cutting-edge technologies, exemplified by acoustic monitoring and automated analysis, represents a paradigm shift towards achieving these goals. These advanced methods not only enhance accuracy and efficiency but also of-

fer real-time capabilities, ensuring a more comprehensive and timely assessment of manatee populations.

The use of audio recordings has emerged as a pivotal method in monitoring manatee populations, introducing a non-invasive and technologically advanced approach to counting individuals[1]. Manatees, known for their unique vocalizations encompassing various sounds, offer a rich source of information crucial for communication, social interactions, and environmental awareness. Deploying hydrophones in their habitats enables researchers to tap into this acoustic landscape, providing a non-invasive means to gather critical data. The continuous, 24/7 surveillance afforded by audio recordings addresses limitations associated with traditional methods, offering insights into manatee behavior around the clock, including nocturnal activities and responses to environmental changes. The non-invasive nature of acoustic monitoring aligns with ethical considerations, avoiding physical contact or disturbances that may impact the well-being of manatees. Individual recognition through acoustic signatures allows for a more specific understanding of population dynamics, including social structures, migration patterns, and habitat preferences. Technological advancements, particularly in automated sound analysis and machine learning[3], further enhance the rationale for audio recordings by facilitating efficient processing of large datasets, ensuring accurate and scalable population assessments.

2.2. Related work

Various techniques and methodologies have been employed to detect manatee vocalizations from audio recordings, reflecting the dynamic landscape of technological innovations in marine mammal research. One common approach involves sound analysis utilizing signal processing algorithms. Several studies have used this approach, where they take advantage of two main characteristics of manatee vocalization, the strong harmonic content, as shown in Figure 2.1 and the slow decaying autocorrelation function[11]. A traditional methodology that has been established in the detection of manatee vocalizations consists of first implementing a denoising algorithm that is capable of attenuating the background noise, then a detection stage is implemented to quickly discard audio segments that do not contain vocalizations and finally a classification algorithm is responsible for taking the audios detected and classifying them into manatee vocalizations and non-manatee vocalizations.

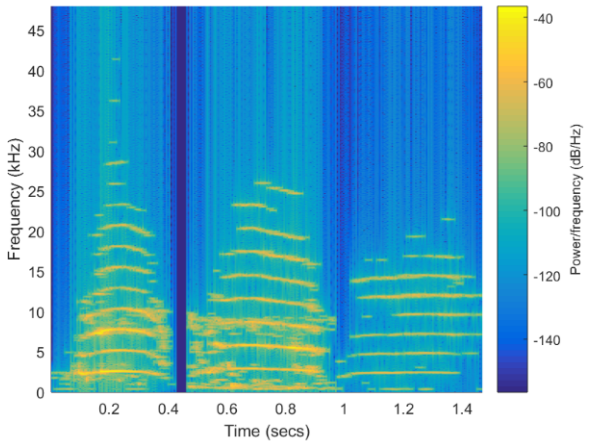


Figure 2.1: Three examples of different manatee vocalizations (Castro and Rivera 2015).

Merchan et al. (2019) introduces a method for automatically detect manatee individuals in continuous passive acoustic underwater recordings. The process involves four stages: detection, denoising, classification, and manatee counting with identification through vocalization clustering. For out interest, we will only go deeper on the first three stages. In this methodology, the detection stage utilizes a modified version of Gur's denoising algorithm (2007), conducting a multi scale signal analysis to identify harmonic and sub-harmonic components based on specific criteria, based on the autocorrelation function, passband filters and duration of the vocalizations. For denoising stage, they proposed a signal subspace approach that is based in decomposing the vector space of the noisy signal into a signal subspace and a noise subspace. The decomposition is obtained using the Karhunen-Loeve transform (KLT). For denoising the signal, the noise subspace is removed by projecting the noised signal in a filtering matrix consisting of components obtained by the KLT of this signal. A modified version of the harmonic detection method, originally proposed by Niezrecki et al. (2003), has been developed for the classification stage. In the original method, the fundamental frequency of vocalizations is estimated by analyzing peaks in the FFT spectrum, and detection is confirmed if at least two harmonics are present. The modified version introduces two additional criteria. Firstly, it verifies that the amplitude of the FFT spectrum in a specified percentage of the frequency band between harmonic components is below a given threshold, accounting for possible subharmonic components. Secondly, when only one harmonic is present (as in some manatee vocalizations), it checks that the amplitude of the FFT spectrum in its vicinity is also below a specified threshold, as indicated by Williams (2005). The results of this method has an average recall of 60 %.

Castro et al (2015) also used the standard methodology, and for the denoising method the undecimated discrete wavelet transform and the autocorrelation function were implemented. Initially, the noisy signal undergoes high-pass filtering at 2 kHz to remove noise outside the vocalization bandwidth. Subsequently, the Daubechis-8 family Undecimated Discrete Wavelet Transform (UDWT) with four decomposition levels is applied to 3 ms windows of the signal. Autocorrelation functions are computed for wavelet coefficients at each level, and Root Mean Square (RMS) values are derived from lag $\tau = 20$ to $\tau = 120$ samples to distinguish between slow and fast decaying envelopes, indicative of manatee calls and noise, respectively. Finally the resulting RMS matrix is smoothed using a 22-point (80 ms) moving average filter to mitigate noise transients. The proposed detection algorithm relies on a matched filter and statistical information concerning fundamental frequency (F0) and peak frequency (Fp). The detection algorithm is shown in Figure 2.2 and involves segmenting the denoised signal using 40 ms Hanning windows with 50 % overlap, where each segment is scored based on its similarity to a manatee call, with the score ranging from -1 to 1. The scoring process considers factors such as the RMS of the signal and the FFT magnitude to identify the peak frequency Fp in the range 2–12 kHz. Fp is assumed to be a harmonic of F0, and candidates for F0 are selected within the range of 1–6 kHz. Each F0 candidate is assigned a score based on the median prominence of up to ten harmonics below 25 kHz.

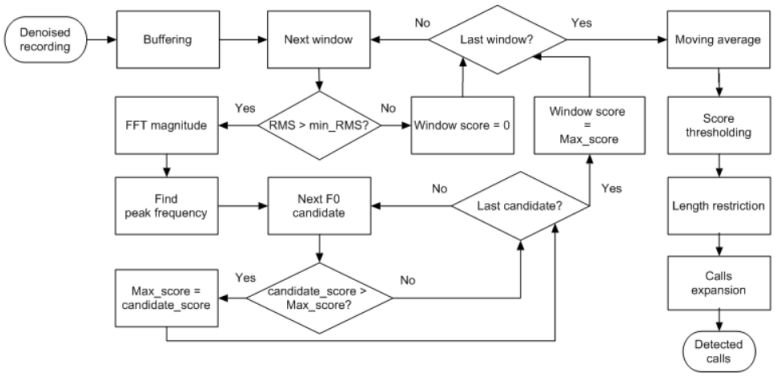


Figure 2.2: Block diagram of the detection method (Castro and Rivera 2015).

A moving average is applied to scores, and a fixed threshold detects manatee vocalizations while filtering out sequences of inadequate durations. The denoising algorithm aids in isolating manatee calls by silencing their surroundings and its output is a binary signal indicating manatee call occurrences through sequences of ones.

This work (Castro et al. 2015) concludes the importance of using a denoising algorithm before doing the detection, since the precision of manatee vocalization classification varies from 10 % to 97 % if the first stage is not used.

Another study carried out by Merchan et al. (2020) propose a comprehensive scheme for identifying and counting manatees using underwater passive recordings, aiming to enhance population estimates in Panamanian wetlands. The four-stage methodology includes detection, denoising, signal classification, and individual counting and identification through vocalization clustering. The denoised algorithm uses Boll's spectral subtraction method[12] to minimize the presence of noise or unwanted artifacts in signals where vocalizations were present. This denoising method had a significantly lower computational cost that the signal subspace approach used previously[2]. The detection stage was the same used in his former work[2]. In this work, for the first time, deep learning techniques are introduced for the classification stage. To do this, the vocalizations detected from the previous stages are transformed in a time-frequency representation, specifically through spectrograms, to later be classified using Convolutional Neural Networks (CNN). To create the spectrograms, they used the FFT-based short-time Fourier transform with 50 % overlapping windows containing 1024 samples. This window size was chosen to get a balance between temporal and frequency resolution suitable for the 96 kHz sampling frequency. Regardless of signal duration, zero-padding and centering were applied to ensure uniformity, resulting in spectrograms with a fixed size of 257 $\times$ 150 pixels. In this study the authors tried different architectures for the CNN with different types of spectrogram generation, and got average recalls of 88 % with precision nearly about 95 %.

2.3. Theoretical framework

2.3.1. Fourier Transform

Signal processing techniques play a pivotal role in analyzing and manipulating signals to extract valuable information. One of the fundamental and widely employed methods in

signal processing is the Fourier Transform, a mathematical operation that decomposes a signal into its frequency components. This technique, named after Joseph Fourier, has become a cornerstone in various scientific and engineering applications due to its ability to provide insights into the frequency domain of a signal.

The Fourier Transform essentially allows us to represent a signal in terms of its frequencies components, offering a powerful tool for understanding and manipulating signals in both continuous and discrete domains. In digital signal processing, where signals are discretized into digital samples, the Discrete Fourier Transform (DFT) is employed and is particularly useful in applications like telecommunications and image processing. One of the most efficient algorithms for computing the DFT is the Fast Fourier Transform (FFT), which dramatically reduces the computation time compared to direct computation, making it suitable for real-time applications.

Fourier transform represents a signal in the frequency domain, revealing the amplitude and phase information of various sinusoidal components that make up the original signal as shown in Figure 2.3.

$$\text{Discrete Fourier Transform (DFT):} \quad X[\omega] = \sum_{n=0}^{N-1} x[n] e^{-i\omega n} \tag{2.1}$$

The mathematical representation of the DFT is shown in equation 2.1, where we can notice that each Fourier coefficient (left side of the equation) is a sum of complex sinusoids, since eulers formula indicates $e^{ix} = cos(x) + isen(x)$. In eq. 1.1, N is the total samples of a discrete signal, $x[n]$ is the amplitude of the signal in the specific sample n and f is the frequency component being analyzed. Here, $X[f]$ represents the amplitude and phase of the k-th frequency component in the discrete signal $x[n]$. The sum considers all samples of the signal, effectively breaking it down into its frequency constituents.

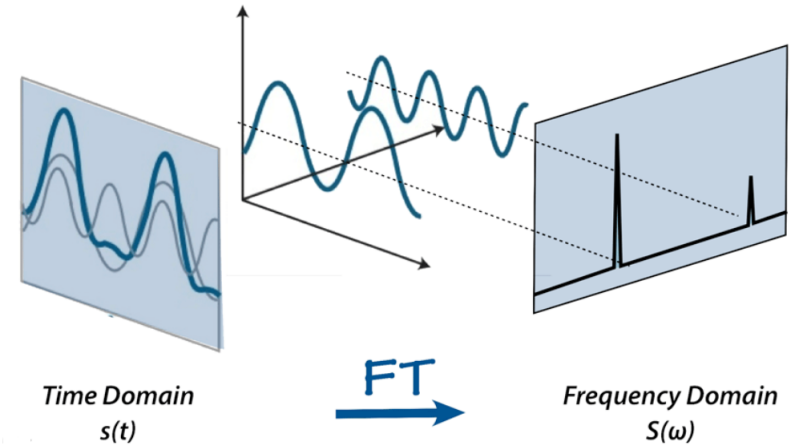


Figure 2.3: Signal decomposition by Fourier Trasform.

Interpretation of the Fourier Transform:

- **Amplitude Spectrum**: the magnitude of $X(f)$ indicates the strength or amplitude of each frequency component in the signal.
- **Phase Spectrum**: the phase of $X(f)$ provides information about the phase relationship

between the different frequency components.

- **Frequency Components**: each term in the Fourier Transform corresponds to a sinusoidal waveform at a specific frequency. The integral or sum evaluates the contribution of each frequency to the signal.
- **Complex Exponentials**: the use of complex exponentials in the Fourier Transform allows for the representation of both sine and cosine components, as well as their phase relationships.

A Specific type of FT is the *Short time Fourier transform* (STFT). This is a signal processing technique that provides a time-varying frequency analysis of a signal. It is an extension of the traditional Fourier Transform, allowing us to examine the changing frequency content of a signal as it evolves over time. This is particularly useful for analyzing signals with non-stationary characteristics, where the frequency components may vary with time.

The STFT is defined by applying the Fourier Transform to short, overlapping windows of the signal. Instead of analyzing the entire signal at once, the signal is divided into segments, and the Fourier Transform is computed for each segment. The use of overlapping windows helps capture the evolution of frequency components across adjacent time intervals.

Mathematically, the continuous STFT of a signal $x(t)$ is given by equation 2.2 and its discrete version on equation 2.3.

$$\mathbf{STFT}\{x(t)\}(\tau,\omega) \equiv X(\tau,\omega) = \int_{-\infty}^{\infty} x(t)w(t-\tau)e^{-i\omega t}\,dt \tag{2.2}$$

$$\mathbf{STFT}\{x[n]\}(m,\omega) \equiv X(m,\omega) = \sum_{n=0}^{N-1} x[n]w[n-m]e^{-i\omega n} \tag{2.3}$$

For the discrete case, but for the continuous is the same logic, $X[m,k]$ represents the STFT at time index m and frequency ω, $x[n]$ is the discrete signal, $w[n-m]$ is a window function centered at time index m, and N is the total number of samples. This algorithm have some key features important highlight, firstly there is a time-frequency tradeoff that allows a compromise between time and frequency resolution. While short windows provide better temporal resolution but poorer frequency resolution, longer windows provide better frequency resolution but poorer temporal resolution. Secondly the choice of the window function is critical in shaping the characteristics of the STFT and overlapping used between windows helps to mitigate spectral leakage and provide a smoother transition between adjacent time intervals. Lastly, the result of the STFT is often represented as a Spectrogram, a two-dimensional plot with time on one axis, frequency on the other, and color intensity indicating the magnitude of the frequency components. This algorithm is very used in task like speech recognition, music analysis and identifying patterns in audio signals. In recent years, the application of the Fourier Transform has expanded with advancements in technology. Real-time processing, particularly in audio and video streaming, relies heavily on efficient Fourier Transform algorithms. Additionally, the integration of machine learning techniques with signal processing has led to innovative approaches in signal denoising, classification, and feature extraction. An example of Spectrogram was shown in Figure 2.1 of previous section.

2.3.2. Deep Learning and Deep Neural Networks

Since last 2 decades Deep Learning, with its Deep Neural Networks (DNNs), have revolutionized the field of IA, learning intricate patterns and excelling other techniques in tasks like image recognition and natural language processing (NLP), leveraging depth for complex representation learning. The basis of deep learning lies in the utilization of artificial neural networks (ANN), drawing inspiration from the structure and functioning of the human brain, consisting of multiple layers of interconnected nodes or neurons, where each of them processes and transforms input data, allowing it to automatically learn intricate features and representations from raw data. The perceptron is the simplest form of a neural network unit, which is an algorithm for learning a binary classifier called a threshold function, a function that maps its input $\mathbf{x}$ (a real-valued vector) to an output value $f(\mathbf{x})$ (a single binary value). The perceptron is a foundational building block in neural networks, capable of learning linear decision boundaries and forming the basis for more complex neural network architectures.

$$f(x) = \theta(\mathbf{w} \cdot \mathbf{x} + b) \tag{2.4}$$

Equation 2.4 shows the perceptron, where θ is the step-function, $\mathbf{w}$ is a vector of real-valued weights, $\mathbf{w} \cdot \mathbf{x}$ is the dot product $\sum_{i=1}^{m} w_i x_i$, where m is the number of inputs to the perceptron, and b is the bias. The bias shifts the decision boundary away from the origin and does not depend on any input value.

2.3.2.1. DNN structure, layers, and activation functions.

Deep Neural Networks (DNNs) are the backbone of modern artificial intelligence, renowned for their ability to learn complex patterns in diverse tasks. As shown in Figure 2.4, a DNN is characterized by its layered architecture, typically comprising an input layer, one or more hidden layers, and an output layer. Each layer consists of interconnected nodes, or neurons, and is responsible for transforming the input data through a set of learnable parameters, known as weights.

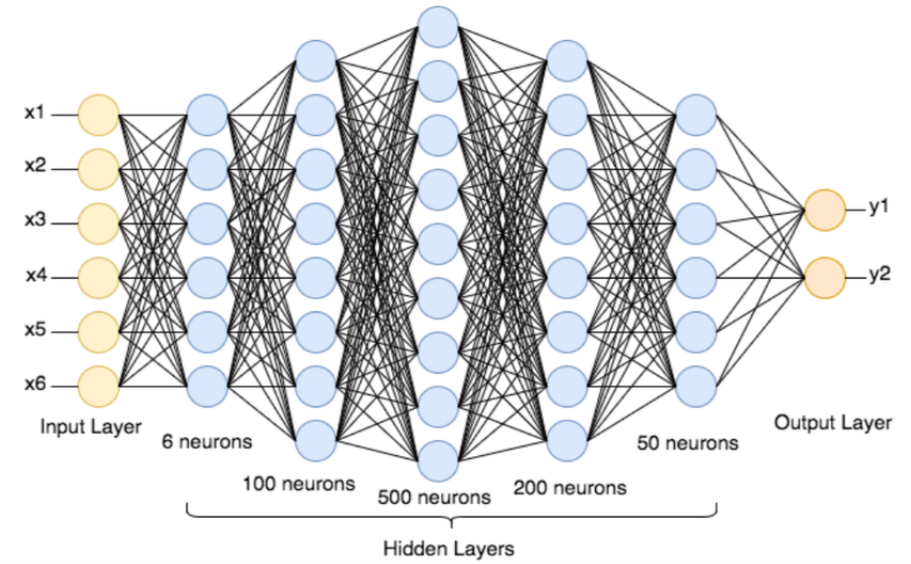
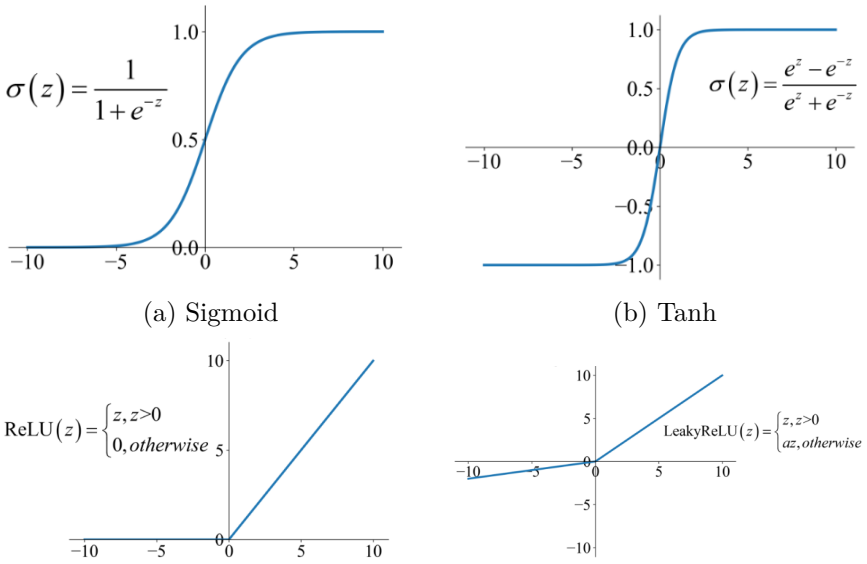


Figure 2.4: Example of a deep neural network arquitecture[13].

The input layer serves as the entry point for data, with each node representing a feature or attribute. The subsequent hidden layers play a crucial role in learning hierarchical representations of the input. These layers leverage activation functions to introduce non-linearities into the model, enabling it to capture intricate relationships in the data. The depth of a DNN, determined by the number of hidden layers, allows it to automatically extract and represent increasingly complex features, enhancing its capacity to understand and generalize from diverse datasets.

Activation functions are a fundamental component of DNNs, introducing non-linearities that enable the model to learn and approximate complex mappings between inputs and outputs. Common activation functions include the sigmoid, hyperbolic tangent (tanh), and Rectified Linear Unit (ReLU). Sigmoid and tanh functions squash the output between 0 and 1 or -1 and 1, respectively, and ReLU, on the other hand, replaces negative values with zero, promoting faster convergence during training and mitigating the vanishing gradient problem. See Figure 2.5 to get a visual interpretation of these activation functions.



(a) Sigmoid (b) Tanh

(c) ReLU (d) LeakyReLU(a=0.2)

Figure 2.5: Commonly used activation functions[14].

The choice of activation function depends on the nature of the task and the characteristics of the data. ReLU has gained popularity for its simplicity and effectiveness in promoting sparsity, but it may suffer from the "dying ReLU"problem where neurons can become inactive during training. Variants like Leaky ReLU address this issue by allowing a small gradient for negative values.

A simple mathematical representation of a neural network of only to layers is presented in equations 2.5, 2.6, 2.7 and 2.8.

$$Z^{(1)} = X \cdot W^{(1)} + b^{(1)} \tag{2.5}$$

$$A^{(1)} = \sigma(Z^{(1)}) \tag{2.6}$$

$$Z^{(2)} = A^{(1)} \cdot W^{(2)} + b^{(2)} \tag{2.7}$$

$$\hat{Y} = \sigma'(Z^{(2)}) \tag{2.8}$$

Where:

- X is the input.
- $W^{(1)}$ and $b^{(1)}$ are the weights and bias of the first layer.
- σ is the activation function.
- $Z^{(1)}$ is the weighted sum for the first layer.
- $A^{(1)}$ is the activation of the first layer.
- $W^{(2)}$ and $b^{(2)}$ are the weights and bias of the second layer.
- $Z^{(2)}$ is the weighted sum for the second layer.
- σ' is the activation function of the output layer, which may vary from the activation functions of the hidden layers.
- $\hat{Y}$ is the predicted output.

The outputs of the neural network are then compared against the actual labels of the examples using a loss function, which is chosen depending on the nature of the problem. The prediction error is computed and then through optimization algorithms the error is propagated through the network in order to adjust its parameters in the direction of minimizing the loss error.

2.3.2.2. Training methodologies and common challenges

Training Deep Neural Networks (DNNs) is a critical phase in realizing their potential, involving the adjustment of model parameters to learn meaningful representations from data. Several methodologies and techniques are employed in this process, along with challenges that researchers and practitioners continually address.

Training key concepts

We present most important concepts and practices for training deep neural networks.

- **Backpropagation**: is the most fundamental training technique for DNNs. It involves the iterative optimization of model parameters by computing gradients of the loss with respect to each parameter and adjusting them accordingly. This process is typically facilitated by optimization algorithms like stochastic gradient descent (SGD) or its variants (e.g., Adam, RMSprop), which determine the magnitude and direction of parameter updates.
- **Mini-Batch Training**: instead of processing the entire dataset in one go, mini-batch training involves dividing the data into smaller subsets. This accelerates training by allowing for more frequent weight updates, enhancing convergence and often providing computational efficiency.
- **Regularization Techniques**: to prevent overfitting, regularization techniques such as dropout or L1/L2 regularization are commonly employed. Dropout randomly deactivates neurons during training, introducing a form of ensemble learning, while L1/L2 regularization adds penalty terms to the loss function to discourage large weights.

- **Data Augmentation**: data augmentation involves applying random transformations to the training data, such as rotation, flipping or masking, to increase variability to the dataset. This helps the model generalize better to unseen examples.

- **Transfer learning**: transfer learning is a technique where a model pretrained on one task is adapted for a related task. It leverages knowledge gained from the source task to improve performance on the target task, especially when datasets share underlying patterns. Approaches include feature extraction, using pretrained model features as input, and fine-tuning, adjusting model weights for the target task. Transfer learning is valuable when target task data is limited, providing a way to benefit from knowledge acquired on a larger source task.

Common Challenges

In the other hand, there are many typical challenge when facing the training of a neural network model. We now present the most relevant ones.

- **Vanishing and Exploding Gradients**: in deep networks, gradients can diminish (vanish) or explode during backpropagation, making it challenging for the model to learn effectively. Techniques like careful weight initialization and using activation functions that mitigate this issue, such as ReLU, aim to address these challenges.

- **Overfitting**: DNNs are prone to overfitting, where the model performs well on the training data but poorly on unseen data. Regularization techniques, proper dataset splitting for validation, and early stopping are strategies used to mitigate overfitting.

- **Computational Intensity**: deep networks often require substantial computational resources for training, making them computationally intensive and time-consuming. Strategies such as distributed training, model parallelism, and hardware acceleration (e.g., GPUs, TPUs) are employed to address these challenges.

- **Hyperparameter Tuning**: selecting optimal hyperparameters, such as learning rate, batch size, and network architecture, is a non-trivial task. Grid search, random search, and more advanced optimization techniques are used to find suitable hyperparameter configurations.

- **Data Quality and Quantity**: the success of DNNs is highly dependent on the quality and quantity of training data. Insufficient or biased data can lead to poor generalization. Data augmentation and careful curation of diverse datasets are strategies to mitigate these challenges.

- **Interpretability and Explainability**: the inherent complexity of deep networks often makes them difficult to interpret. Understanding and explaining the decisions made by these models is an ongoing challenge, especially in sensitive domains like healthcare and finance.

Addressing these challenges requires a combination of domain expertise, algorithmic advancements, and continuous research. As DNNs become increasingly integral to various applications, refining training methodologies and overcoming challenges contribute to the ongoing evolution of deep learning.

2.3.3. Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) represent a specialized class of Deep Neural Networks designed to excel in processing grid-like data, most notably images. They have become pivotal in the realm of computer vision, offering a transformative approach to feature extraction and pattern recognition. Unlike traditional neural networks, CNNs leverage convolutional layers to automatically learn spatial hierarchies within the input data, making them particularly effective in tasks where the spatial arrangement of features is crucial.

The fundamental building blocks of a CNN are convolutional layers, pooling layers, and fully connected layers, as can be seen in Figure 2.6. Convolutional layers employ filters or kernels that slide over the input data, capturing local patterns and creating feature maps. Pooling layers downsample the spatial dimensions of the feature maps, reducing computational complexity while retaining important information. Fully connected layers at the end of the network aggregate high-level features for classification or regression tasks.

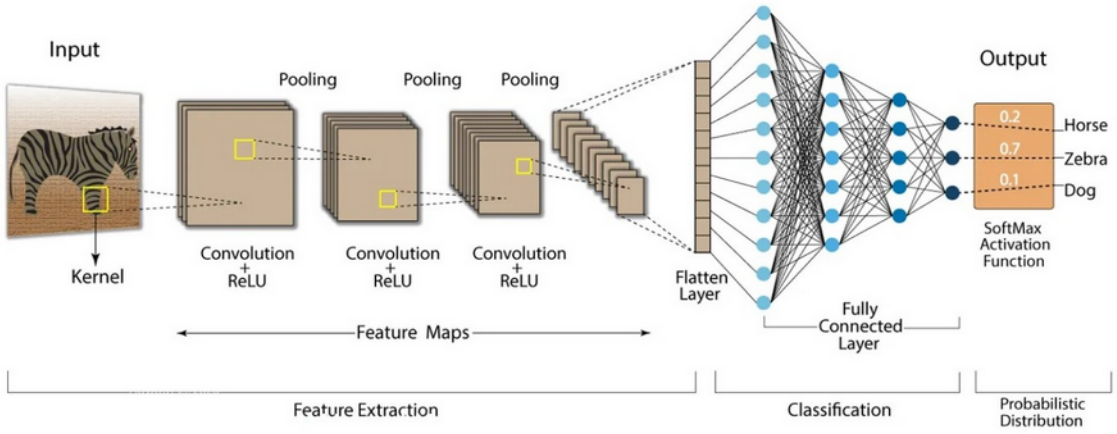


Figure 2.6: Example of a convolutional deep neural network arquitecture[15].

Convolutional layers apply filters or kernels to local regions of the input data, allowing the network to automatically learn and extract spatial features. These filters slide over the input, performing element-wise multiplications and aggregating the results to create feature maps. Through multiple convolutional layers, the network progressively learns abstract and hierarchical representations, capturing complex patterns and structures within the data. The use of non-linear activation functions, such as ReLU, introduces non-linearities and enhances the model's expressive power.

In the other hand, pooling layers are interleaved between convolutional layers and are crucial for spatial down-sampling. Max pooling and average pooling are common techniques, reducing the spatial dimensions of the feature maps while retaining essential information. Pooling helps make the network more computationally efficient, reduces overfitting, and enhances translation invariance, making the model robust to variations in object positions within the input.

Some of the main applications of CNN architectures are described below:

1. **Image Classification**: CNNs have revolutionized image classification tasks, surpassing traditional methods by automatically learning hierarchical features. Applications range from identifying objects in everyday photos to classifying medical images for diagnostic purposes.

2. **Object Detection**: CNNs are widely employed in object detection tasks, accurately localizing and classifying objects within images. Popular architectures like YOLO (You Only Look Once)[16] and Faster R-CNN[17] have become benchmarks in this domain, finding applications in autonomous vehicles, surveillance, and more.

3. **Semantic Segmentation**: for tasks requiring pixel-level precision, such as medical image analysis or autonomous navigation, CNNs are employed for semantic segmentation. They assign a specific label to each pixel, delineating object boundaries within an image.

4. **Facial Recognition**: CNNs power facial recognition systems, enabling applications like biometric authentication in smartphones and surveillance systems. They learn intricate facial features, making them robust to variations in pose, lighting, and facial expressions.

2.3.4. Transformers

2.3.4.1. Overview of transformers and their attention mechanism.

Transformers represent a groundbreaking architecture in deep learning, introduced in the seminal paper *Attention is All You Need* by Vaswani et al. (2017). Their innovative design departs from traditional sequential processing methods, offering a parallelized approach that excels in capturing long-range dependencies in sequential data. Originally designed for natural language processing tasks like machine translation, transformers have since proven to be versatile and applicable across various domains due to their capacity to model complex relationships in data[18]. The big picture of Transformers arquitecture is shown in Figure 2.7.

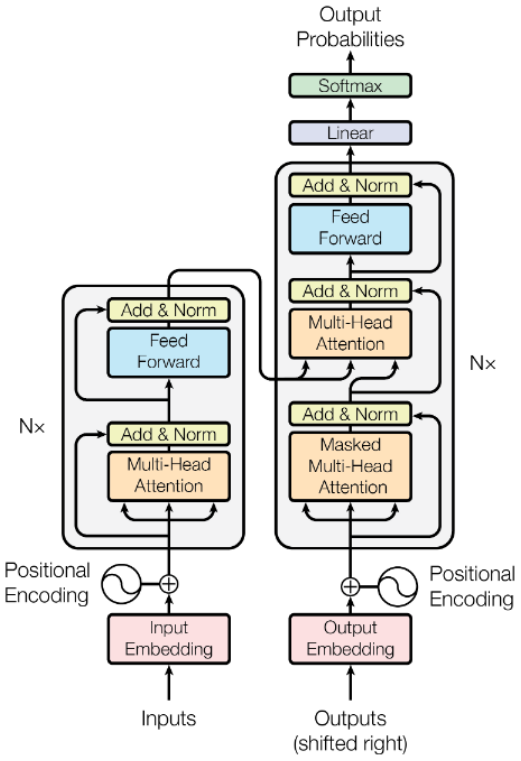


Figure 2.7: Transformer arquitecture[19].

At the heart of the transformer architecture lies the attention mechanism, a fundamental concept that revolutionized the way models process and contextualize information, first introduce by Bahdanau et al. (2014)[20]. The attention mechanism enables the model to selectively focus on different parts of the input sequence, assigning varying degrees of importance to each element. An illustrative example of this is shown in Figure 2.8. This selective attention allows transformers to capture dependencies that span across the entire sequence, overcoming the limitations of fixed-size receptive fields present in traditional recurrent neural networks (RNN).

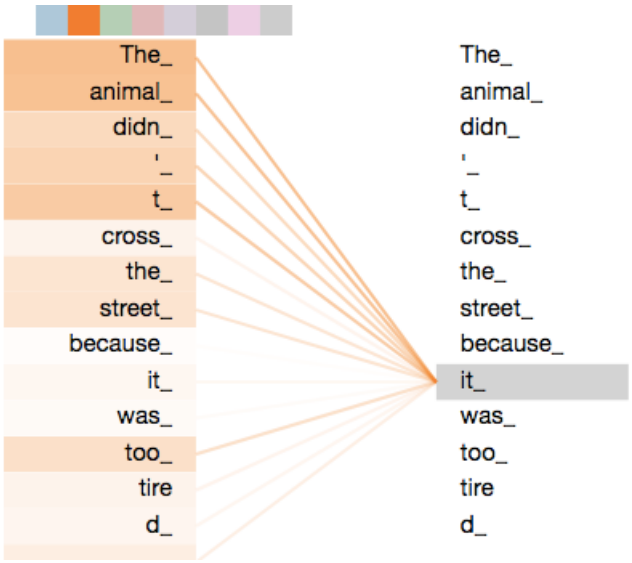
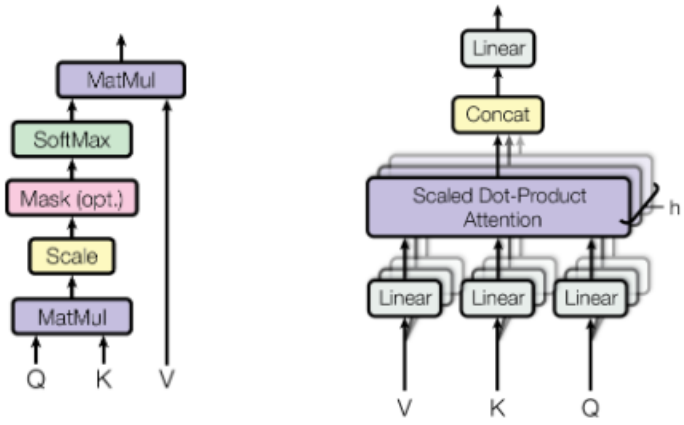


Figure 2.8: Illustrative example of how attention mechanism works for a NLP task[21].

The attention mechanism in transformers can be broadly divided into two types, Self-Attention or Scaled Dot-Product Attention, and Multi-Head Attention, see Figure 2.9. In the first mechanism, the model weighs the importance of different tokens in a sequence concerning each other. The attention score is computed by taking the dot product of the **querys** (Q), representing elements in the input sequence for which the model seeks context, **keys** (K), that store information about each element for reference, and **value vectors** (V) that store relevant information about the context, providing a weighted sum of values for each position in the sequence. This selective attention enables the transformer to capture intricate dependencies in a concise and expressive manner. In the other hand, Multi-Head Attention mechanism is used to enhance the expressive power of attention mechanisms. In this configuration, the model learns multiple sets of attention weights in parallel, allowing it to capture different aspects of relationships within the data. The outputs from different attention heads are concatenated and linearly transformed to produce the final attention-weighted representation.



(a) Self attention. (b) Multi-Head attention.

Figure 2.9: Types of Transformer attention mechanism.

The attention mechanism facilitates the modeling of complex dependencies, enabling transformers to excel in tasks like machine translation, text summarization, and question answering. Moreover, the parallelized nature of transformers makes them highly efficient for training on modern hardware, contributing to their widespread adoption in the deep learning community.

2.3.4.2. Vision Transfomer (ViT)

The Vision Transformer, or ViT, is a neural network architecture designed for computer vision tasks. It is based on the Transformer architecture, motivated by the success of Transformers in NLP, making researchers to explore their application in other domains. The big picture of its architecture is presented in Figure 2.10.

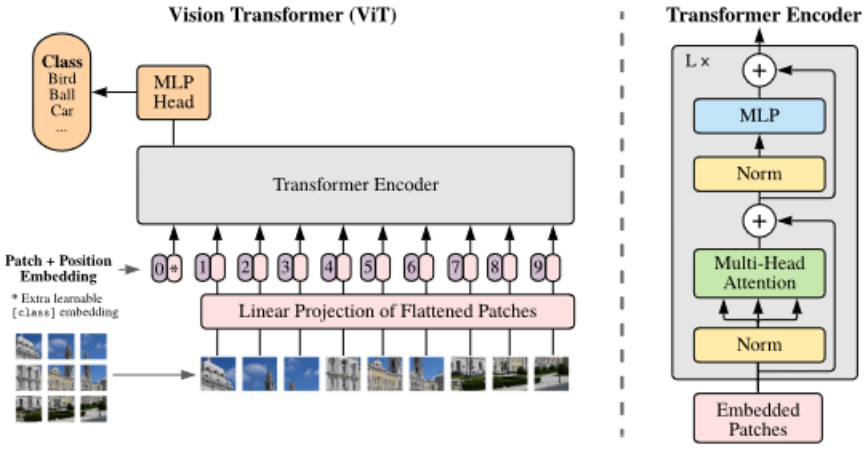


Figure 2.10: ViT arquitecture[18].

Decomposing the ViT architecture into it main components.

- **Patch Embedding**: the input image is divided into fixed-size non-overlapping patches, where each patch is linearly embedded into a lower-dimensional vector. This process converts the 2D image into a sequence of embeddings, similar to how words are embedded

in NLP tasks.

- **Positional Encoding**: since the Transformer architecture doesn't inherently understand the spatial relationships between tokens, positional information is crucial. Positional encodings are added to the patch embeddings to provide information about the position of each patch in the original image.
- **Transformer Encoder**: the core of the Vision Transformer is the Transformer encoder architecture. Self-attention mechanisms are used to capture global dependencies between different patches, allowing each patch to attend to all other patches, capturing long-range dependencies.
- **Classification Head**: the final output of the transformer encoder is used for classification. A simple classification head, often a fully connected layer, is added to predict the class labels.

This deep neural network benefits from global context understanding, making it suitable for tasks that require understanding relationships across the entire input, such as image classification. Also ViT can scale effectively to handle both small and large images, capturing complex patterns when train on large datasets[22]. In addition, as other large models, it can be pretrained on large datasets and then be fine-tuned on smaller, task-specific datasets, leveraging the knowledge learned and adapt it to specific visual task. Vision Transformers have demonstrated impressive performance on various computer vision benchmarks and competitions, establishing them as a powerful architecture in the field of computer vision tasks[18].

2.3.4.3. Audio Spectrogram Transformer (AST)

Audio Spectrogram Transformer (AST) is a Vision Transformer adapted to work with audio instead of images. The essence of AST its the same as ViT, because it does not use the audio as a waveform representation, but first transform it into Spectrograms that represent the time-frequency content of audio signals, breaking it down into its frequency components over time. This is a 2D representation of the audio, that can be treated as a 1 channel image, which is exactly what AST model does. The AST architecture is shown in Figure 2.11.

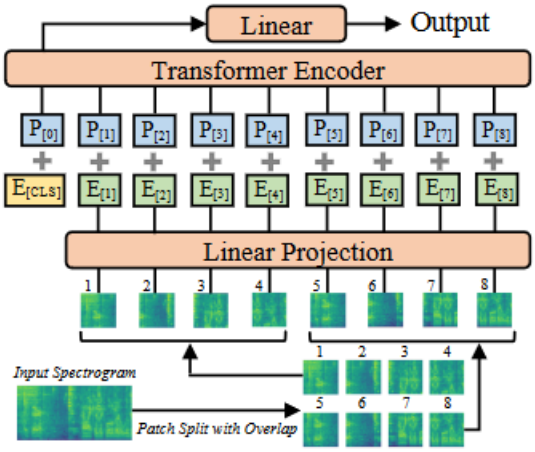


Figure 2.11: AST arquitecture[22].

AST ans ViT are similar models, sharing the same components described in the ViT

section, but AST was designed for audio-related tasks, working in the domain of audio signals, making it suitable for tasks like speech recognition, sound classification, and audio generation.

While both AST and ViT share the underlying Transformer architecture, their applications and specific design considerations are tailored to the characteristics of audio and visual data, respectively. These models showcase the versatility of the Transformer architecture across different modalities.

Chapter 3

Objectives

3.1. General objective

Develop a methodology to detect manatee vocalizations in audio segments, ensuring adaptability for audios recorded in different environments, through the integration of signal processing and deep learning techniques.

3.2. Specifics objectives

- Conduct a comprehensive literature review on existing methodologies for audio analysis, signal processing, and deep learning techniques in the context of detecting manatee vocalizations in audio segments.
- Develop a methodology that can adapt it solution to ensure reliable performance across different recording settings.
- Evaluate the performance of the developed methodology through rigorous experiments, using a labeled dataset of manatee vocalizations.
- Present a detailed analysis of the methodology's strengths, weaknesses, and potential areas for future improvements.

Chapter 4

Methodology

In this study we propose to use a novel approach that allows coupling the training of the denoise model with the classification model so that the parameters of both models are adjusted under the same objective, which is to be able to correctly predict the existence of manatee vocalizations in an audio segment. To do this we create a model that we will call *joint model*, which will have three main components. The first component is a denoise model, which will be responsible for removing the background noise from the audio segments. This model is based on the **power spectral floor denoising** algorithm proposed by Tobar et al. (2021). The second stage will take this denoise audios and transform them into a time-frequency representation, using for this the *Short Time Fourier Transform*. Finally we pass this representations to a classification model, which is the last and third component of the joint model. The model we propose to use for the classification stage is the *Audio Spectrogram Transformer*[22]. We can see the proposed methodology in Figure 4.1.The following sections details the implementation of these three sections, along with the dataset used and its preprocessing.

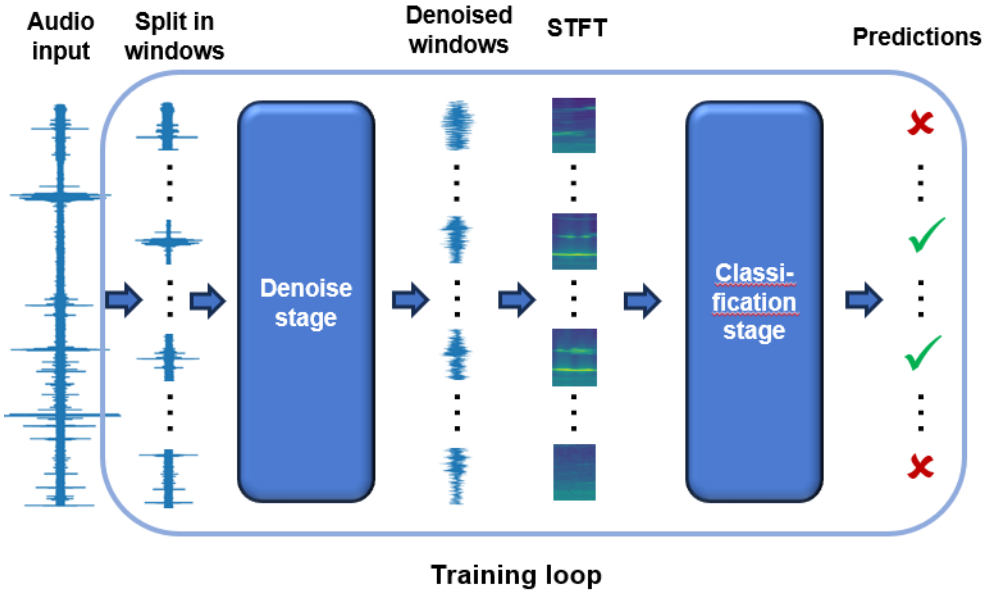


Figure 4.1: Proposed methodology.

4.1. Dataset and preprocessing

Our study considered 20 approximately 10-minute-long audio sessions, registered in Zoo-Parc de Beauval (Saint-Aignan, France), with a total of 3 hours of labelled data, collected over a three-week period from November 2020 to January 2021. The sampling rate of the recordings was 48 kHz and for each audio session there was annotations indicating all the manatee vocalizations registered with their starting and ending time. Each recording contained between 3 and 52 manatee vocalizations, and represent less than 1 % of total data. Manatee vocalizations are in average 240 [ms] long, with a range from 100 [ms] to 600 [ms]. The audio recordings where obtained using an omnidirectional hydrophone (Aquarian Audio, H2A-XLR, sensitivity of -180dB re: 1V/μPa, frequency range response 20 Hz to 100 kHz,) with a Zoom H5 recorder (24-bit quantization and 48 kHz sampling rate; recording level was set manually to 80).

Before feeding the model with this recordings, we first split each audio file into windows of 100 [ms], with an overlap of 50 [ms]. The length of the chosen window had to be large enough to contain the stationary part of a manatee vocalization and short enough in order not to contain non-stationary data. Additionally, we assigned a single label for each window and positive labels where consider when a 100 % of the vocalization was inside the windows intervals. This segmentation gave us a total of 218.702 samples, where only 0.5 % of them were from the positive class (containing manatee vocalization). The total dataset generated was split in a train, validation and test set, each of them with 60 %, 20 % and 20 % of the total examples respectively.

4.2. Denoise Stage

The denoise stage aims to minimize the presence of noise or unwanted sounds in signals where vocalizations were present. We used **power spectral floor denoising** proposed by Tobar et al. (2021), consisting on removing the *power spectral floor* of the recording. The power spectral floor is calculated as the power spectrum (the squared magnitude of the Fourier transform) of a typical noise window. This noise window is find as the 25th percentile by ordering the windows with respect their maximum power spectrum value. However in our implementation we don't fix the 25th percentile, but rather we learn it during training. The model is the following:

$$F_{denoised}(x) = F(x) * \sigma(\alpha * \frac{P(x) - \beta * P_{floor}}{\beta * P_{floor}}) \tag{4.1}$$

where:

- x is each audio segment window.
- $F(x)$ is the Fourier transform of the recording x.
- $P(x) = ||F(x)||^2$ is the Power spectrum of x.
- P_{floor} is the power spectral floor window.
- α control the smoothness of the sigmoid function.
- β control the scaling of the power spectral floor.

Another difference with Tobar et al. (2021) work is that they tuned by experimentation the values of α and β hiperparameters, using 2.5 and 50 respectively, while we proposed that they also should be learned during training. This distinction allows these parameters to be adjusted to improve the classification performance of the joint model. As seen in Figure 4.1, this model takes noisy audio segments as input and delivers the same segments as output but denoised.

4.3. Generation of time-frequency representation of denoised audio segments

After noise is removed from each audio segment, the time-frequency representation of each window is computed. We choose the *Short Time Fourier Transform* (STFT) as the time-frequency representation for the signals. This is a different approach used by Gong et al. (2021) for AST model, since they used Mel filterbank features. We propose using STFT instead of Mel filterbanks for two reasons, the first is that manatee vocalizations have frequencies above 2 [kHz] that when using the Mel scale begin to compress since it gives greater resolution to the low frequencies and lower resolution to high frequencies, since the human does not perceive the different frequencies in a linear way but rather distinguishes the low frequencies better over the high ones. The second reason is because manatee vocalizations have harmonic components up to 20 [kHz], whose relationship in the frequency dimension is lost when transformed to the Mel scale. To compute the STFT we use 70 % overlapping Hanning windows of 256 samples. The choice of the window size was influenced by the decision to use the pretrained weights of the AST model. Since Gong et al. (2021) used 128 log mel filterbanks features, we need to get a resolution of 128 frames in the frequency domain, which is achieve using windows of 256 samples. Also, mention that we use the log power of the spectrum to create the STFT for numerical stability and regardless we got 55 bins in the time dimension, they were zero-padded to obtain spectrograms with a fixed size of 128 $\times$ 128. This may not seem reasonable because it does not add useful information, but during experimentation it proved to have better performance since it takes more advantage of the pre-trained weights of the AST, since it uses an input of 128x1024. Lastly, we normalized this STFT to be 0 mean and 0.5 std, as recommended by Gong et al. (2021) when using their pretrained weights for AST model.

4.4. Classification stage

For this stage we test *Audio Spectrogram Transformer* (AST) (Gong et al. 2021) as the classification model. In this model, initially the t seconds input audio waveform, is converted into a 128-dimensional log Mel filterbank (fbank), computed at 10 [ms] intervals using a 25 [ms] Hamming window. This yields a $128 \times 100t$ log mel spectrogram that serves as input for the AST. The log mel spectrogram is then divided into N 16×16 patches with strides of 10 frames in both time and frequency dimensions, establishing the number of patches and the effective input sequence length for the Transformer. Each 16×16 patch is flattened into a 1D patch embedding of size 768 through a linear projection layer referred to as the patch embedding layer. This linear projection is made with a 2D convolutional layer. As the Transformer architecture lacks the ability to capture input order information, and the patch sequence is not in temporal order, a trainable positional embedding of size 768 is added to each patch

embedding. This addition enables the model to grasp the spatial structure of the 2D audio log mel spectrogram. AST also include a [CLS] token at the sequence's start, as the ViT model does, which is a learnable embedding serving as the spectrogram representation. Given that AST is tailored for classification tasks, only the Transformer's[19] encoder is utilized. Importantly, the original Transformer encoder architecture is adopted without alterations. This choice is deliberate for two reasons: 1) the standard Transformer architecture is straightforward to implement and replicate as it is readily available in TensorFlow and PyTorch, and 2) to facilitate transfer learning in AST. Specifically, the Transformer encoder employed has an embedding dimension of 768, 12 layers, and 12 heads, mirroring those in a reference work [18]. The output of the [CLS] token in the Transformer encoder serves as the representation of the audio spectrogram and to map ot to classification labels, a multi layer perceptron with sigmoid activation is applied. We modified the original version of this AST model, as mention in the previous section, to adapt it to work with our data. The modification where only made on the generation of the time-frequency representation, where instead of 128 log mel filterbanks features used by the original work, we used the STFT also with 128 frames in the frequency dimension. This is the only adaption made to the AST arquitecture.

4.5. End-to-End training framework

As is identified in Figure 4.1 our methodology proposal implies training simultaneously the denoised model with the classification model, so both sets of parameters are adjusted through the same pipeline. This way the denoise model updates its parameters in order to improve the classification task which is the main objective. For these purpose we couple the denoise model, the generation of the spectrogram and the classification model in one big model which we will call the *joint model*. This model takes as input the audio segments coming from the preprocess stage and outputs the predictions whether it is a vocalization (positive class) or not (negative class).

We had to adapt some of the hyperparameters of AST model to work with our dataset because Audioset dataset consists in 10 [s] long audios, which are 100 times longer than our segments of audio of 100 [ms], so the log mel spectrogram for Audioset has around 1000 time frames (using 25 [ms] as window length and 10 [ms] of stride) where for our dataset, using same window length and window stride, we get around only 8 time frames. Considering that, we instead of using audios with a 16 [kHz] sampling rate, which give us audio segments of 1600 samples, we use the original sampling rate of the recordings of 48 [kHz], getting audio segments of 4800 samples, but keeping the window length and stride according to initial 16 [kHz] frequency rate. We also decrease the window stride when computing spectrogram to half, 5 [ms], to get more resolution in temporal dimension considering the small size of our audio segments. Doing this we get a time-frequency representation of audios with 128 bins in frequency dimension and 55 in time dimension.

We used pretrained weights of the AST learned over Audioset dataset[23], to transfer the knowledge acquire for the model, since to training from scratch we would need millions of examples as mentioned by Gong et al. (2021).

We also had to restrict the *percentile* parameter of the denoiser model, as it have intrinsically interpretation and has to be between 0 and 1, so we passed through a sigmoid function Considering the unbalanced dataset we have, where positive samples represent the 0.5 % of the all dataset, we used a weighted sampling strategy during training by assigning weights to each sample, influencing their probability of inclusion in training batches. Higher weights in-

crease the likelihood of a sample being chosen. This is particularly useful when certain classes are underrepresented, ensuring that the model learns from all classes effectively. By drawing samples with probabilities based on their weights, the model is exposed to a representative mix of classes, mitigating bias and enhancing generalization.

In addition we used a linear learning rate scheduler to dynamically adjusts the learning rate during the training process. A fixed learning rate may lead to suboptimal training outcomes or slow convergence. A learning rate scheduler addresses these issues by allowing the model to adapt its learning rate based on the progression of training. Initially, a higher learning rate facilitates faster convergence, while later in training, a smaller learning rate helps fine-tune the model and avoid overshooting optimal parameter values. This adaptability enhances training stability, accelerates convergence, and ultimately contributes to better generalization and model performance.

Lastly we also used a warm-up strategy for the learning rate which is useful in the early stages of training neural networks to address challenges related to model instability and convergence. It avoid an .[ea]rly over-fitting"problem which may occur when the dataset is highly differentiated and if happens to include a cluster of related, strongly-featured observations. If it the case, the model's initial training can skew badly toward those features. Warm-up is a way to reduce the primacy effect of the early training examples. Without it, you may need to run a few extra epochs to get the convergence desired, as the model un-trains those early superstitions.

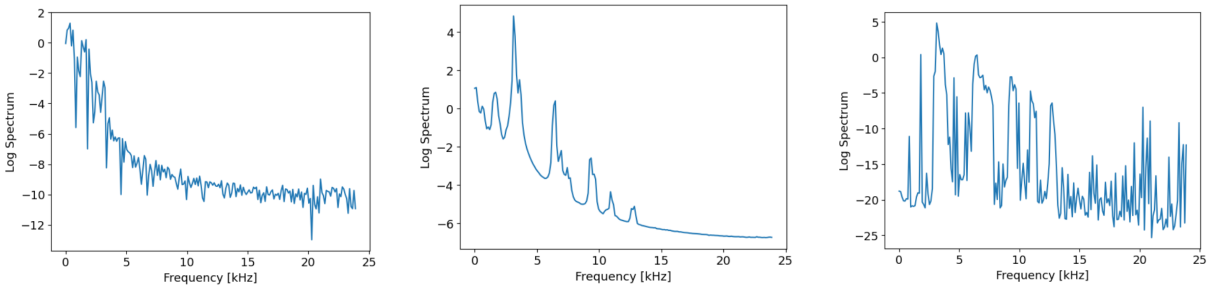
Chapter 5

Experiments and results

In this section we present the experiments carried out with their respective results. We first test independently the denoising and the classification model, to verified if they accomplish what they are suppose to do. To evaluate the performance of the classification model we use the recall and precision metrics. They were observed separately, since the first allows us to know how many of the total manatee vocalizations the model is detecting, and on the other hand the second metric gives us the Look at how accurate you are being with your predictions. However, to peak the best model we use F1-score since is the harmonic mean between recall and precision. We trained the models in a remote server with a Intel(R) Xeon(R) CPU E5-1620 v3 @ 3.50GHz, with a NVIDIA GeForce GTX 1080 GPU. Each run of 30 epochs took about 20 hours

5.1. Denoising results

To test the denoising model by it self we first find the parameters α, β and *percentile* by experimentation, starting by the values used in Tobar et al. (2021) and changing them in order to effectively remove background noise and enhance the manatee vocalization. In Figure 5.1 we can see in a) the log spectrum of a typical noise window in our audio recordings and in b) an average of log spectrum of windows containing vocalizations, where we can see the harmonic frequency components present in manatee vocalizations. This results were obtained using $\alpha = 10$, $\beta = 100$ and $percentile = 0.25$.



(a) Typical noise window log power spectrum.

(b) Manatee vocalization log power spectrum.

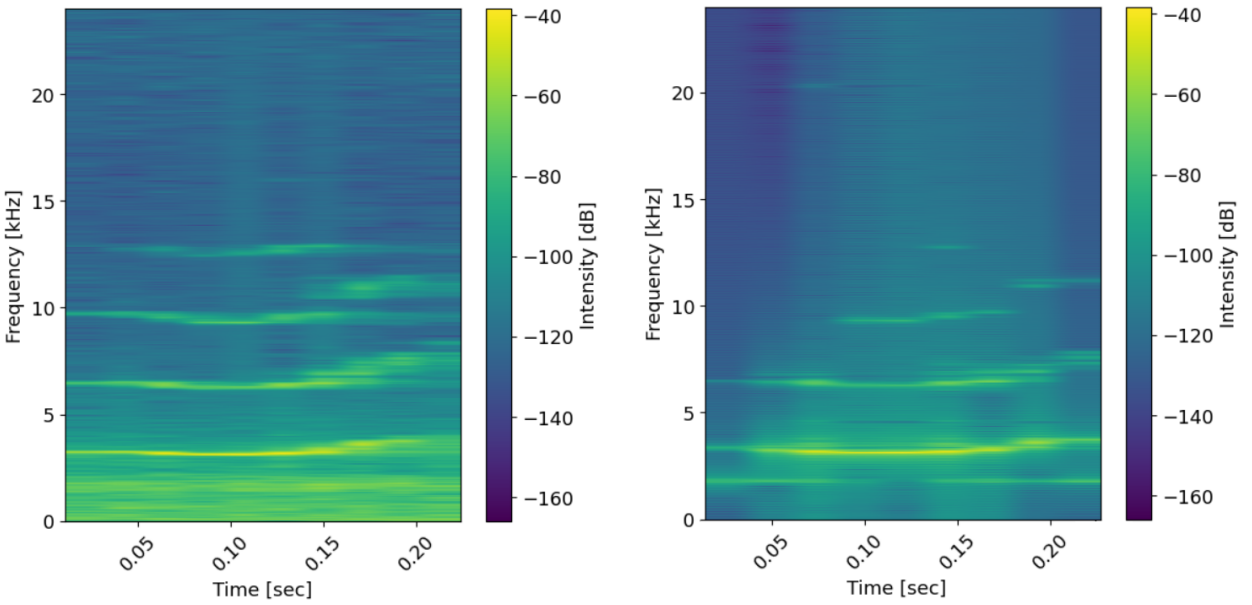
(c) Denoised manatee vocalization log power spectrum.

Figure 5.1: Log power spectrum of a typical noise window, a manatee vocalization and a denoised manatee vocalization.

We can observe that noise has low frequency range, contrasting with manatee vocaliza-

tions where they has a bandwidth much broader. Now applying the denoise algorithm to the manatee vocalization, i.e. subtracting the spectrum of the noise window to the audio recording, we can observe the result in image c) of Figure 5.1 where it is noticed that frequencies components of the noise window are subtracted from the vocalization.

To get a more expressive representation of the performance of the denoise algorithm Figure 5.2 shows a) the spectrogram of a noisy vocalization in contrast of its b) denoised version.



(a) Noisy vocalization.

(b) Denoised vocalization.

Figure 5.2: Spectrogram of a noisy and denoised vocalization.

We can see that frequencies below 2.5 [kHz] are attenuated, keeping harmonics components of the vocalization and therefore enhancing them, which is exactly the objective of the denoise model.

5.2. Classification results

On the other side, we also tested the classification model by itself to verify firstly if can detect manatee vocalizations from denoised audios. As mentioned in the methodology section, we had to tune some of the architecture of AST to be able to work with our data. Need to mention that we used the pretrained AST weights learned during training with Audioset dataset[23] because the size of our dataset is considerably small to train such a big model as AST with approximately 87 millions of parameters. This transfer learning is very straightforward just loading the available weights which are accessible thanks to the author and taking care of the normalization of our input to be 0 mean and 0.5 std. We used the denoised audio segments obtained as the output of the denoise model as input of the AST. The results are shown in Table 5.1.

Table 5.1: Classification model tested independently

Model name	**Recall**	**Precision**	**F1 score**	**Accuracy**
Only AST	77.3 %	73.7 %	75.5 %	99.1 %

This results shows us that the model is learning to discriminate between vocalizations and not vocalizations and tell us that we picked a correct model for this task.

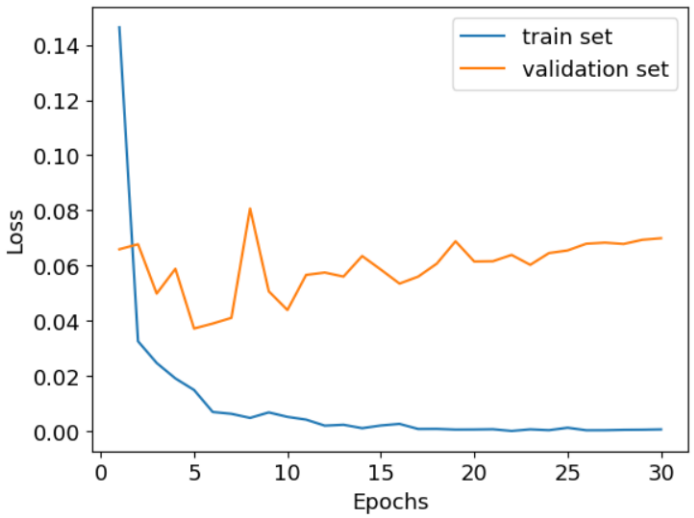


Figure 5.3: Training and validation loss evolution for AST independent.

As seen in Figure 5.3, during the training it seems that the model has some overfitting, so there is space to improve its performance if we apply some data augmentation techniques or we get more data.

5.3. Joint model results

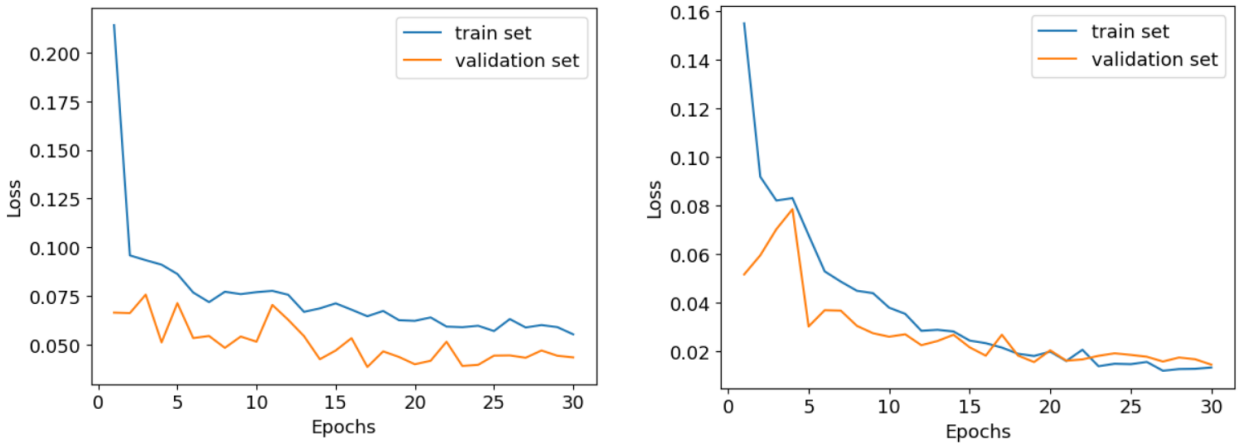
After testing both models independently we couple them in one pipeline of training, so the can update both parameters at the same time. We tested four different training configurations, which emerged as a consequence of the results obtained from the previous runs. Each training set up and hyperparameters used are the following.

Table 5.2: Hyperparameters

Variable	**Value**
Learning rate	$3e^{-5}$
Epochs	30
Optimizer	Adam
Input dimension	128x128
Loss	Binary cross entropy
Batch size	64

In the first training loop (experiment #1) we implemented the joint model with priors on the parameters of the denoise model, initializing them with the same values we found in the independent testing stage of the model and using them as a starting point. The results presented in Table 5.3, shows that this joint model performs better than the independent AST model, specially when we look at the recall metric, which indicates is detecting roughly 13 % more manatee vocalizations. In image a) of Figure 5.4, we can see the evolution of the

loss during training and unlike the previous model now it seems that there is still more space for improving the performance since the train set loss is decreasing, so probably running the model for more epochs will end with better results. Also important to remark is that the validation loss is evolving below the train loss, this could be due to the regularization used in the model.



(a) Joint model.

(b) AST only with noisy inputs.

Figure 5.4: Examples of loss evolution during the epochs for validation and training sets.

After training, the parameters of the denoise model were observed and these did not suffer major variations and remained with values very close to those with which it was initialized. We consider two possible options, the first is that since the classification model is very large (87 million parameters) the gradient vanishes and is not able to reach the denoiser through back propagation, causing its parameters to not fit. The second option is that the initial parameters were actually close to an optimal value, so it would be necessary to vary them greatly.

To verify the first option, we ran another training loop (experiment # 2) with a new model created with skip connections from the denoise model directly to the MLP that generates the classification on AST model, in order to see if the parameters of the Denoiser do fit in this way. To do this, was passed the Denoiser parameters as inputs in the forward pass of the AST model, which are concatenated at the end before entering the MLP that generates the prediction. In this way a direct connection is created from the Denoiser to the end of the joint model that allows the gradient to flow through this new path directly to the Denoise model.

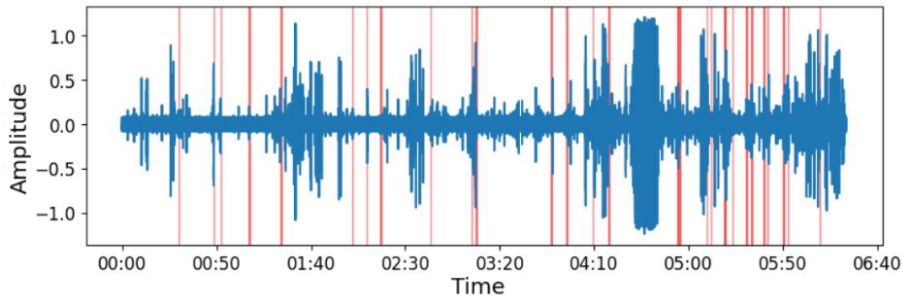
Table 5.3: Models performance

Model name | Exp	**Recall**	**Precision**	**F1 score**	**Accuracy**
Only AST | #0	77.3 %	73.7 %	75.5 %	99.1 %
Original joint model | *#1*	91.0 %	70.8 %	79.6 %	99.2 %
Skip connections | #2	61.4 %	82.9 %	70.5 %	99.1 %
Random initialization | #3	88.5 %	71.3 %	78.9 %	99.1 %
AST only with noisy input | #4	85.7 %	80.2 %	82.9 %	99.4 %

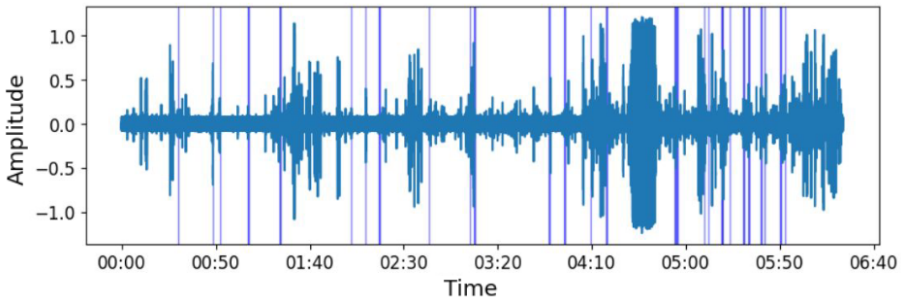
The results in Table 5.3 shows that the model has worse performance than the previous ones, but yet when looking at the Denoiser parameters we can still see that they remain practically unchanged from their initial values. With this results the option that the parameters are not being adjusted because the gradient is not capable of reaching begins to be loose arguments, and the second option that the parameters with which the denoise model was initialized are close to the optimal parameters and for this reason they do not change, it begins to be more plausible. To test that (experiment # 3), the weights of the Denoise model are initialized with random values, and the model is retrained, without the skip connections, in order to see if the parameters do vary in this way. The results in Table 5.3 show that the model continues to perform quite similar to the model of the first experiment, and the weights of the Denoise model still do not change much with respect to their initial values.

This make us think that it seems the model is agnostic to the values that the denoise parameters take when generating the predictions. We could see, for example, that the beta parameter that is responsible for scaling the typical noise window went from 100 to 1 and yet the model did not seem to be affected. This opens the possibility that the model may not need a denoise model for this particular problem, and this may be because the audios were recorded in a zoo, where the noise is much lower than in a natural habitat (e.g. sea, river), perhaps noise is not a determining factor for the correct detection of manatee vocalizations. To test the above, we will experiment by directly passing the audios with noise to the AST model (experiment # 4). The result for this model, shown in Table 5.3, says that there is no need to use denoised audios to be able to distinguish between positive and negative samples. Is the model with better performance, if we use F1 score as the comparison metric. As usual seen in other results, this model with noisy audio inputs have better recall than precision, which indicates its power to detect manatee vocalizations. The evolution of its training and validation loss, is shown in Figure 5.4 image b), and we can observe that there is also space for improving performance as it seems is still decreasing and is not overfitted yet.

In Figure 5.5 we show an illustrative example of the detected samples of the joint model for the audio session recording number 4.



(a) Ground truth vocalizations.



(b) Detected vocalizations.

Figure 5.5: Illustrative example of the detected vocalizations of the joint model for session 4.

Chapter 6

Discussion and future work

The results presented in the section leave us a lot to talk about and analyze. First of all, as mentioned en previous sections, the recall and precision metrics are the ones that interest us the most because on the one hand, the first indicates the capacity of the model to identify manatee vocalizations and the second shows us how precise it is, when it indicates that a sample is of the positive class. This is extremely relevant because if we focused only on precision, the model could tend to be very careful when predicting to do not make any error, yet be able to recognize only few of the total manatee vocalizations, which would be a big problem. On the other hand, if it were the other way around and we paid attention only to the recall, the model would ensure and predict a large part of the samples as a positive class, getting high detection rate at the cost of poorer precision. This tradeoff between recall and precision is well demonstrated in Figure 6.1, which shows how these metrics evolve during training, taking the training of the joint model as an example.

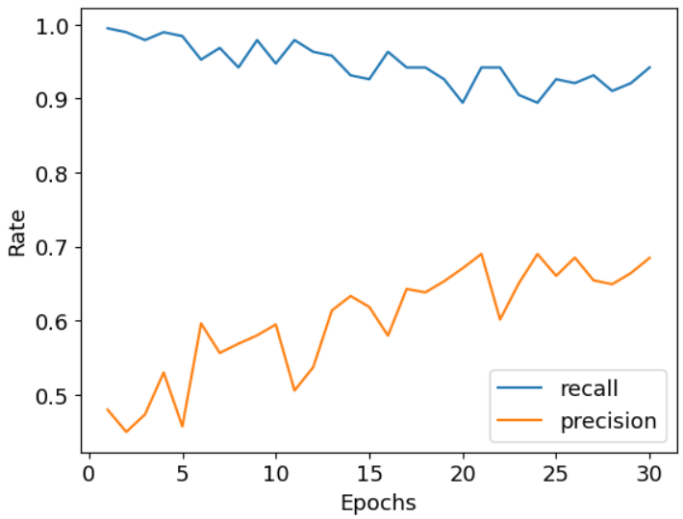


Figure 6.1: Evolution of precision and recall for the joint model training.

Considering the above, when observing the results presented in the previous section, it is worth mentioning that all models, except the model combined with skip connections, are capable of recognizing over 77 % of all manatee vocalizations. This is a significant achievement considering that only 0.5 % of the total samples correspond to this class, which could be a major obstacle for the model to adjust its parameters to recognize positive or negative examples effectively. On this point, the combined model (experiment #1) has the highest

recall, exceeding 90 %, but at the same time, it makes more mistakes when predicting a sample as a manatee vocalization, as it has the lowest precision. Continuing in the same line, it is noteworthy that when looking at this metric, we can observe that the model with the best performance is precisely the one with the worst recall, i.e., the model with skip connections. This may occur because adding three extra dimensions to the input of the final layer of the AST model, the MLP classification layer, gives more expressiveness to the classification, making it better at predicting the positive class but simultaneously reducing its detection power.

To balance and weigh both metrics of interest, we look at them together through the F1 score. Regarding this metric, we can observe that the model with the best performance is the AST model, whose inputs are segments of noisy audio without having undergone the denoising process. While this model may not stand out the most when looking at recall or precision individually, it performs the best in combination, which is precisely what we are seeking, a model that detects a significant portion of vocalizations while being precise when doing so. These results, indicating that the model with noisy inputs performs best in F1 score, raise a lot of questions for discussion. The first thing that can be said about this is that perhaps the AST model, trained with the Audioset dataset containing segments of noisy audio, learned to be capable of identifying patterns in noisy inputs and when we provide it with segments of audio without noise, we might be hindering the model's performance because it encounters samples from a distinct distribution than the one it was trained on. This is despite the fact that the samples were normalized before being input into the model. Another possibility is that the denoiser is removing relevant information from manatee's vocalizations, as can be seen in Figure 5.2, where, when comparing the noisy vocalization (a), some of the harmonic components of the vocalization are lost when the noise is removed (b).

In the other hand, when looking at the accuracy metric, which indicates the total number of samples correctly identified, considering both positive and negative samples, all models show high performance. However, this is not a metric of great interest for our specific problem, since the dataset is highly imbalanced with a ratio of 1/200 in terms of positive to negative samples and models could predict everything as the negative class and still have a high accuracy, which is not our case.

Table 6.1: Variation of denoise model parameters.

Model name | Exp	α	β	**Percentile**
Original joint model | #1	0.23 %	0.62 %	0.1 %
Skip connections | #2	0.46 %	0.82 %	0.1 %
Random initialization | #3	0.17 %	0.53 %	0.2 %

Discussing the results obtained from the different training sessions of the joint model, experiments #1, #2, and #3, where it was observed that the parameters of the denoise model did not vary much from their initial values (see Table 6.1), leads us to think that these parameters may not be of great help for the model when learning the representations of the inputs. That's why their values are not adjusted, which becomes more plausible when we observe that the AST model with noisy inputs performs better than the combined model with the denoise model. Another possibility is that the learning rate used during training is very low (as recommended by Gong et al. 2021), and given that the size of the AST model

is very large, this could be causing issues when updating the weights of the denoiser.

The last point to mention is that perhaps the denoise model is also eliminating certain frequency components of manatee vocalizations that impair their correct detection. This does not happen when using noisy audios, where the vocalization is fully preserved without adding or removing any information from it.

6.1. Future work

This work is still ongoing, and the results presented in this report are those obtained up to the current date. However, there are many things to address after having observed and discussed the results presented in the previous section. First of all, what needs to be done is to experiment with different hyperparameter configurations during training sessions, as a single combination has been used for all trained models so far. Regarding the training setup, firstly, we need to train for more epochs since, as seen in Figure 5.4, there is room for improvement for some of the presented models, since losses are still decreasing till last epoch. Extending the training of these models could enhance their performance and we would simply need to take the learned weights up to epoch 30 and train again the models from this point. On the other hand, as observed in Figure 5.3, some models seem to have overfit, given that the validation loss diverges from the training loss. To address this, we could include more data (currently in the process of collection) and also implement data augmentation techniques to introduce variability into the dataset. One of the data augmentation techniques applied by Gong et al. (2021) in their training is spectrogram masking, where masks are added to portions of the spectrogram, in both the temporal and frequency dimensions, to introduce variability to the dataset and force the model to learn robust representations of the inputs.

Another thing we are going to do is vary the learning rate during training, as this may be affecting the update of the denoiser's parameters. We will also modify the learning rate scheduler, which reduces the rate by half every 5 training epochs, resulting in a learning rate of the order of magnitude of $1e^{-8}$ at epoch 30. Additionally, during experimentation, we tried varying the dimensions of the input for the AST model, specifically the dimensions of the spectrogram, and we extended the temporal dimension to 1024, following the approach used by Gong et al. (2020), and the results improved compared to the 128x128 inputs we are currently using. This is noteworthy because, by definition, our inputs are 1000 times shorter than those used to train the AST model. Therefore, when we mention extending the input dimension, what we are essentially doing is padding with zeros up to frame 1024. Intuitively, this might not make much sense as we are not adding extra information to the smaller spectrogram. However, the results seemed promising. Unfortunately, they are not presented here as we could only run them for 5 epochs due to slow training, because the input became to large, and due to computational constraints, we could only use a batch size of 2 samples, making the training process time-consuming.

On the other hand, although the combined model demonstrated worse performance than the AST-only model, this could be because the audios were recorded in a zoo, and background noise may not be an obstacle to learning to discriminate between classes. However, if we test with audios recorded in the natural habitats of manatees (rivers, seas), perhaps the combined model improves compared to the AST-only model. This because providing the model with audio segments without noise, using data in real habitats, could be beneficial for the classification model when making predictions.

Another option that we should try is to train the AST-only model and the combined

model without using the pretrained model provided by the author. This should not result in an improvement in performance since it requires too much data to update its 87 million parameters effectively, however, it is still worth experimenting with.

Finally, it is worth mentioning that this work can be extended beyond manatee vocalization detection and use the positive predictions of the model to segment spectrograms using clustering algorithms, aiming to identify different individuals based on the differences in their vocalization frequency components (William et al., 2005). This approach has been pursued by other authors[1–3], and their results have been promising in terms of identifying distinct individuals.

Chapter 7

Conclusions

From the present study, several relevant aspects can be concluded after observing the obtained results. Firstly, although this work is still in progress, it was noted that implementing a combined model, with a denoise model and a classification model coupled in the same training framework, did not yield better results than those obtained by the classification model alone, but they are still good and not so far from the AST-only model. This could be attributed to the dataset used for this work, where the audios were collected in a zoo in France with a well-controlled ambient noise, which may not be a problem for the AST model when generating accurate predictions for manatee vocalization detection. This highlights the capability of the Transformer-based model, which performed better not only compared to its combined version with the denoise model but also when using noisy inputs instead of using inputs that had the noise removed first.

However, it remains to be verified how the results would be when using audio segments collected in the natural habitat of manatees, where ambient noise is much more invasive than in a zoo. If logic holds, there should be a notable difference between using noiseless audios versus noisy ones, since in a natural environment, noisy signals could potentially prevent the correct identification of harmonic components in manatee vocalizations, which should be crucial for the AST model to learn which segments of the spectrogram to pay attention on.

This work demonstrates the tremendous capability of attention-based models to work with spectrograms, representing a state-of-the-art technique that enables the detection of over 80 % of manatee vocalizations while maintaining over 80 % precision, even when positive class examples are only 0.5 % of the total dataset.

While the obtained results do not validate the novel proposal of creating a combined model that trains a denoise model along with a classification model, we have no choice but to continue working to validate the use of a combined model on data from a different source. This will help determine whether the AST model indeed does not require noise-free inputs, as it pretrained weights were learned using noisy data, or if our proposal for joint training is genuinely useful for improving performance in the detection and accurate classification of manatee vocalizations.

Bibliography

[1] Castro, J. M., Rivera, M., y Camacho, A., "Automatic manatee count using passive acoustics", Proceedings of Meetings on Acoustics, vol. 23, p. 010001, 2015.

[2] Merchan, F., Echevers, G., Poveda, H., Sanchez-Galan, J. E., y Guzman, H. M., "Detection and identification of manatee individual vocalizations in Panamanian wetlands using spectrogram clustering", The Journal of the Acoustical Society of America, vol. 146, pp. 1745–1757, 2019.

[3] Fernando Merchan, Ariel Guerra, H. P. H. M. G. J. E. S.-G., "Bioacoustic classification of antillean manatee vocalization spectrograms using deep convolutional neural networks", 2020.

[4] Sagredo, B., Español-Jiménez, S., y Tobar, F., "Detection of blue whale vocalisations using a temporal-domain convolutional neural network", en 2021 IEEE Latin American Conference on Computational Intelligence (LA-CCI), pp. 1–5, IEEE, 2021.

[5] Bowen, W., "Role of marine mammals in aquatic ecosystems", Marine Ecology-progress Series - MAR ECOL-PROGR SER, vol. 158, pp. 267–274, 1997, doi:10.3354/meps158267.

[6] Guzman, H. M. y Condit, R., "Abundance of manatees in panama estimated from side-scan sonar", Wildlife Society Bulletin, vol. 41, no. 3, pp. 556–565, 2017, doi:https://doi.org/10.1002/wsb.793.

[7] Bonde, R., Aguirre, A. A., y Powell, J., "Manatees as sentinels of marine ecosystem health: Are they the 2000-pound canaries?", EcoHealth, vol. 1, pp. 255–262, 2004, doi:10.1007/s10393-004-0095-5.

[8] Ackerman, B., "Aerial surveys of manatees: a summary and progress report", Population Biology of the Florida Manatee. National Biological Service, Information and Technical Report 1. Washington, D.C. 289 pp., p. 289, 1995.

[9] Deutsch, C., Bonde, R., y Reid, J., "Radio-tracking manatees from land and space: Tag design, implementation, and lessons learned from long-term study", Marine Technology Society Journal, vol. 32, pp. 18–29, 1998.

[10] LaCommare, K. S., Brault, S., Self-Sullivan, C., y Hines, E. M., "Trend detection in a boat-based method for monitoring sirenians: Antillean manatee case study", Biological Conservation, vol. 152, pp. 169–177, 2012, doi:https://doi.org/10.1016/j.biocon.2012.02.021.

[11] Gur, B. M. y Niezrecki, C., "Autocorrelation based denoising of manatee vocalizations using the undecimated discrete wavelet transform", The Journal of the Acoustical Society of America, vol. 122, pp. 188–199, 2007, doi:10.1121/1.2735111.

[12] Boll, S., "Suppression of acoustic noise in speech using spectral subtraction", IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. 27, no. 2, pp. 113–120, 1979, doi:10.1109/TASSP.1979.1163209.

[13] Bahi, M. y Batouche, M., "Deep learning for ligand-based virtual screening in drug discovery", pp. 1–5, 2018, doi:10.1109/PAIS.2018.8598488.

[14] Feng, J., He, X., Teng, Q., Ren, C., Chen, H., y Li, Y., "Reconstruction of porous media from extremely limited information using conditional generative adversarial networks", Physical Review E, vol. 100, 2019, doi:10.1103/PhysRevE.100.033308.

[15] Haque, K. N., "What is convolutional neural network — cnn (deep learning)", 2023, https://www.linkedin.com/pulse/what-convolutional-neural-network-cnn-deep-learning-nafiz-shahriar/ (visitado el 03-04-20230).

[16] Redmon, J., Divvala, S. K., Girshick, R. B., y Farhadi, A., "You only look once: Unified, real-time object detection", CoRR, vol. abs/1506.02640, 2015, http://arxiv.org/abs/1506.02640.

[17] Ren, S., He, K., Girshick, R. B., y Sun, J., "Faster R-CNN: towards real-time object detection with region proposal networks", CoRR, vol. abs/1506.01497, 2015, http://arxiv.org/abs/1506.01497.

[18] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., y Houlsby, N., "An image is worth 16x16 words: Transformers for image recognition at scale", 2021.

[19] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., y Polosukhin, I., "Attention is all you need", CoRR, vol. abs/1706.03762, 2017, http://arxiv.org/abs/1706.03762.

[20] Bahdanau, D., Cho, K., y Bengio, Y., "Neural machine translation by jointly learning to align and translate", 2016.

[21] Alammar, J., "The illustrated transformer", 2018, http://jalammar.github.io/illustrated-transformer/ (visitado el 2018-06-27).

[22] Gong, Y., Chung, Y.-A., y Glass, J., "Ast: Audio spectrogram transformer", 2021.

[23] Gemmeke, J. F., Ellis, D. P. W., Freedman, D., Jansen, A., Lawrence, W., Moore, R. C., Plakal, M., y Ritter, M., "Audio set: An ontology and human-labeled dataset for audio events", en 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 776–780, 2017, doi:10.1109/ICASSP.2017.7952261.

Annexes

Annex A. Cálculos realizados

A.1. Metodología

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Figure A.1: Imagen en anexo.

A.2. Resultados

Suspendisse vitae elit. Aliquam arcu neque, ornare in, ullamcorper quis, commodo eu, libero. Fusce sagittis erat at erat tristique mollis. Maecenas sapien libero, molestie et, lobortis in, sodales eget, dui. Morbi ultrices rutrum lorem. Nam elementum ullamcorper leo. Morbi dui. Aliquam sagittis. Nunc placerat. Pellentesque tristique sodales est. Maecenas imperdiet lacinia velit. Cras non urna. Morbi eros pede, suscipit ac, varius vel, egestas non, eros. Praesent malesuada, diam id pretium elementum, eros sem dictum tortor, vel consectetuer odio sem sed wisi.

Table A.1: Tabla de cálculo.

Elemento	ϵ_i	**Valor**	**Descripción**
A	10	$3{,}14\pi$	Valor muy interesante[a]
B	20	6	Segundo elemento
C	30	7	Tercer elemento[1]
D	150	10	Sin descripción
E	0	0	Cero

[a] Este elemento tiene una descripción debajo de la tabla
[1] Más comentarios