TECHNIQUES TO OVERCOME DATA SCARCITY IN DEEP LEARNING FOR PASSIVE ACOUSTIC MONITORING OF MARINE MAMMALS

by

Bruno Padovese

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy

at

Dalhousie University
Halifax, Nova Scotia
September 2023

Table of Contents

List of Tables

List of Figures

Abstract

Passive Acoustic Monitoring (PAM) is a useful technique for monitoring marine mammals. However, the large volume of data collected through PAM systems make automated algorithms for detecting and classifying sounds essential. Deep learning algorithms have shown great promise in recent years, but their performance is limited by insufficient amounts of annotated data for training the algorithms. Our work examines several machine learning techniques to overcome data scarcity in a single and multi-domain scenarios, where each domain is a different underwater acoustic environment. We first investigate the benefits of augmenting training datasets in a single domain with synthetically generated samples when training a deep neural network for the classification of marine mammals. We apply two acoustic data augmentation techniques, SpecAugment and Mixup, on PAM data to improve the network‘s performance. Next, we address the challenge of data scarcity in a multi-domain context through transfer learning, a machine learning concept whereby knowledge from a source domain is transferred to a target domain. Specifically, we considered two different underwater acoustic environments as the source and target domain. We develop a more robust deep neural network model for the classification of marine mammals by incorporating knowledge from two different domains. Lastly, we confront data scarcity in a scenario where no annotated data is available for training deep learning models. In this context, we explore the artificial generation of synthetic marine mammal vocalizations, integrating real acoustic properties from the underwater environment to create datasets for training deep neural networks in detecting and classifying real marine mammal vocalizations. We evaluate the performance of all three approaches and compare the results with baseline models. We demonstrate that the proposed approaches provide useful and effective solutions in scenarios of data scarcity under diverse and variable conditions.

List of Abbreviations

ASR Automatic Speech Recognition.

CLI Command-Line Interface.

CNN Convolutional Neural Network.

DDSP Differentiable Digital Signal Processing.

DL Deep Learning.

DNN Deep Neural Network.

DWT Discrete Wavelet Transform.

EMB Emerald Basin.

FN False Negative.

FP False Positive.

FPR False Positive Rate.

GAN Generative Adversarial Network.

GOM Gulf Of Maine.

GSL Gulf of St. Lawrence.

KNN K-Nearest Neighbors.

LFDCS Low Frequency Detection and Classification System.

MARU Marine Autonomous Recording Units.

MFCC Mel Frequency Cepstral Coefficients.

ML Machine Learning.

NARW North Atlantic right whale.

PAM Passive Acoustic Monitoring.

PE Parabolic Equation.

QBC Query-By-Committee.

ReLU Rectified Linear Unit.

ResNet Residual Network.

RNN Recurrent Neural Network.

SNR Signal-to-Noise Ratio.

STFT short-time Fourier Transform.

SVM Support Vector Machine.

TFBD Time-Frequency-Based Detection.

TL Transfer Learning.

TP True Positive.

Acknowledgements

The methods and algorithms implemented are a contribution of the Canadian Foundation for Innovation MERIDIAN cyberinfrastructure (https://meridian.cs.dal.ca/).

Chapter 1

Introduction

Many marine mammals primarily utilize species-specific acoustic signals for communication and navigation within their environment [34]. In light of this, passive acoustic monitoring (PAM) has gained widespread acceptance for studying animal vocalizations (bioacoustics) in both aquatic and terrestrial environments [133], proving to be a valuable tool in research, conservation efforts [16], species distribution estimation [48], population size determination [86], and other applications [84, 93]. Furthermore, PAM is non-invasive and eliminates the risk of behavioral alterations that may result from other monitoring techniques, such as GPS tagging [136].

With advancements in PAM technology and reduced costs, autonomous recording units have been deployed to cover extensive areas, and capable of continuous data collection for extended periods (ranging from months to years) [25]. However, the development of effective tools for analyzing recordings to detect marine mammal vocalizations has not kept pace with the improvements in acoustic data collection methods. In many cases, manual analysis supported by traditional signal processing techniques [123, 38] remains the primary method for examining and annotating these datasets. This approach often demands highly specialized expertise and can be both difficult and inefficient. Consequently, the creation of fully or nearly fully automated detection and classification systems for target signals is desirable.

In the past few years, Machine Learning (ML) algorithms have been incorporated into many of these systems due to their potential to efficiently process large amounts of data and accurately identify signals of interest [65, 127]. In particular, Deep Learning (DL), a subfield of ML, has shown substantial progress in discriminative tasks across several fields [49, 91, 97], with a number of recent studies dedicated to the detection and classification of marine mammal vocalizations such as beluga whale signals [149], sperm whale clicks [11] and dolphin echolocation clicks [79].

One key advantage of DL over traditional ML techniques is that it reduces the

need for a pre-processing pipeline to convert raw audio data into features for detectors and classifiers [5, 21, 105]. Traditional feature extraction often involves application-specific, complex steps that require expertise in multiple domains [8, 13]. In contrast, DL incorporates the feature extraction process into a Deep Neural Network (DNN) architecture, learning data representations through a multi-layer approach. Moreover, DL models can be adapted and tuned to new applications by simply providing them with new data. This simplicity allows DNNs to be highly adaptable to different classification tasks [104], and enables them to outperform conventional ML techniques by producing more useful representations than the traditional process of manual feature engineering [128]. However, in practice, most DL-based detectors and classifiers still employ some pre-processing steps, such as computing spectrograms [139, 127].

Typically DNNs require large amounts of training data to successfully train accurate detectors and classifiers. However, while large amounts of underwater acoustic data have been collected with the help of PAM systems, only a small fraction of these data have been labeled due to the large cost involved in the annotation process. The scarcity of annotated data constitutes the major bottleneck towards developing effective DL models for marine mammal classification. In addition, DL applications to PAM data face some key challenges that can heavily impact a DNN model's ability to generalize well to previously unseen data, particularly in small datasets. One key challenge is that underwater acoustic data is characterized by the surrounding underwater soundscape in which it was collected, and may vary greatly depending on the geographic location that the PAM system was deployed. Furthermore, real world underwater acoustic data is often subject to substantial amounts of anthropogenic noise sources such as shipping and construction activities. These signals often overlap with animal vocalizations and are another source of confusion for a classifier [10]. Another major challenge in underwater bioacoustic datasets is that valuable signals are often strongly underrepresented when compared to the background noise [10]. Therefore, before engaging with new costly and time consuming annotation efforts, we can employ DL techniques, such as data augmentation [128], transfer learning [150], semi-supervised learning [102] and active learning [89], that can help train DNNs that generalize better with smaller training datasets.

In this work, we explore two of these techniques, namely data augmentation and

transfer learning (TL), to address the problem of data scarcity and generalization when training a DNN on PAM data for the detection and classification of the "upcall" vocalization of North Atlantic right whales (NARW; *Eubalaena glacialis*). The NARW is an endangered baleen whale species that has been the subject of intensive research over the past several decades in an effort to conserve the species. The upcall vocalization, a distinctive call produced by NARWs, is frequently the focus of detection and classification methods, [65, 127, 130]. However, while the focus of this work is on NARW upcall vocalizations, the methodologies examined can be easily adapted to accommodate different species and incorporated into other acoustic detection and classification frameworks. In summary, the work can be observed through three distinct phases.

In phase one, we introduce the concept of data augmentation applied to underwater acoustics. Data augmentation is a strategy that can be employed to significantly inflate the size and diversity of the training dataset by generating new labeled data through the manipulation of the current available data. The augmented data are variations of the current available data that were not seen before but are theoretically possible. While particularly effective when there is data insufficiency, augmentation has been applied to great success in larger datasets as a form of regularization to reduce the problem of overfitting in DL models [52]. Overfitting occurs when a model learns too well the details and noise of the training data but fails to predict unseen data. Data augmentation has long been used to improve the performance of DL models across many tasks [70, 49, 67]. Taylor *et al.* [135] showed that even simple geometric and photometric transformations could have a big impact on a DNN performance. Wang *et al.* [111] compares the effectiveness of traditional augmentation techniques to generative methods based on generative adversarial networks (GANs) on relatively small amounts of data. In environmental acoustics, Salamon and Bello [120] use audio data deformations such as time stretching, and pitch shifting to train a DNN model on urban environmental sounds, vastly improving performance over the DNN trained without augmentation. In this research, we propose the utilization of two acoustic data augmentation methods, SpecAugment [108] and Mixup [147] to improve the classification of NARW upcalls. For this purpose, we train a DNN on a dataset comprised of sound clips containing upcall vocalizations and compared the

results with and without the augmentation step. We then evaluate the effectiveness of these two methods in a scenario of data scarcity, that is, when there is an insufficient amount of data to successfully train a DNN.

In phase two, we explore the concept of TL applied to underwater acoustics. As previously discussed, the diversity and variations in underwater acoustic environments along with data insufficiency constitutes a major challenge towards developing a uniform model that generalizes well to PAM data from different sources. Data collected from PAM systems can differ in location, depth and type of the system deployment, resulting in different sources of transient sounds/noise (human activity and other biological sounds), different ambient sounds (noise from breaking waves, wind, rain, etc), differences in sound propagation (a vocalization may sound different due to reverberations, diffraction, attenuation, reflections), differences in hardware and system self-noise, among many other differences. TL can mitigate this problem by applying the knowledge learned in one or more source domains to a different target domain. This approach can be particularly effective as some of the features learned in the source domain often share similarities with the target domain [150]. In this study, we consider the source domain as a DL model that has been trained on a large annotated dataset representing an acoustic environment, and the target domain as a different acoustic environment that we wish to apply the model to. We employ TL to adapt the trained model to the target environment using considerably less annotated data than was originally used to train the model. We then compare the performance of the TL model against a model that was trained only on the data from the target environment.

In phase three, we approach the issue of data scarcity from a scenario where no annotated data is available to train a DNN. Specifically, we consider a large acoustic dataset without any annotations. In this context, traditional data augmentation techniques and TL may be insufficient to mitigate the problem, as they depend on the availability of at least a minimal amount of annotated data for transforming the data, creating new samples, or adapting to a novel environment. To overcome this limitation, we employ artificial generation of synthetic signals by mimicking the acoustic properties of NARW upcalls vocalizations. Initially, we create synthetic signals resembling the shape of the upcall without any distortion effects. Next, we

utilize acoustic propagation modeling to simulate how these synthetic signals would propagate and distort in a real underwater environment. Finally, we leverage the large quantities of unlabeled data available to embed these synthetic signals into segments of real background data, thereby generating a new sample containing information from both the synthetic vocalization and the environmental soundscape. By utilizing both synthetic data and real background environment samples extracted from a large unprocessed dataset, we train a DNN to detect and classify reall NARW upcalls. We compare the results obtained by a model trained on synthetic data with a model trained on real vocalizations, and assess whether this approach can produce a model capable of serving as a useful tool for assisting bioacousticians in analyzing large quantities of PAM data.

1.1 Contributions

In the first phase, we train a DNN for the detection and classification of marine mammal vocalizations and enhance its performance through data augmentation on data from a single source domain. We contribute in the following:

- We present a DNN capable of classifying marine mammal vocalizations upcalls
- We adapt the SpecAugment and Mixup augmentation techniques to PAM data, showing that these techniques lead to a significant improvement in the ability of our DL model to classify Marine Mammal upcalls.
- We demonstrate that data augmentation can be successfully applied when working with very limited amounts of data, greatly boosting model performance while also reducing the required amount of annotated data.

In the second phase, we expand the work done in the first phase to include data from multiple domains (data collected in different underwater acoustic environments), and approach this problem through a TL solution. The contributions of this phase are the following:

- We describe a methodology for adapting a DNN trained at detecting NARW upcalls from a particular set of acoustic environments to a new acoustic environment.

- We evaluate whether the adapted DNN can be used effectively as a detector, extracting a high number of true detections while producing a low number of false detections.

- We implement the methodology as an open-source command-line tool to facilitate the creation of DL-based acoustic detectors and classifiers for NARW upcall vocalizations.

In the third phase, we tackle data scarcity by using artificially generated synthetic signals that mimic marine mammal vocalizations, enabling DNN training on a large, unannotated acoustic datasets, thus overcoming the limitations of the data augmentation and TL approaches of phase one and phase two. Our contributions in this phase are as follows:

- We develop a method for artificially generating synthetic vocalizations that can be used to train a DNN in the absence of annotated data.

- We employ acoustic propagation modeling to simulate the propagation and distortion of synthetic signals in a real underwater environment, enhancing their realism.

- We demonstrate the effectiveness of our approach by training a DNN on synthetic vocalizations, and comparing its performance to a model trained on real vocalizations.

- We assess the potential of this approach to serve as a useful tool for assisting bioacousticians in analyzing large quantities of PAM data, particularly in situations where annotated data is scarce or unavailable.

In addition, this research has contributed to a number of journal publications and invited talks.

1.1.1 Journal Publications

- Padovese, B., Frazao, F., Kirsebom, O. S., Evers, C., Beslin, W. A. M., Theriault, J., & Matwin, S. (2023). Adapting Deep Learning Models to New Acoustic

Environments - A Case Study on the North Atlantic Right Whale Upcall. Ecological Informatics. p. 102169. [107]

- Padovese, B., Frazao, F., Kirsebom, O. S., & Matwin, S. (2021). Data augmentation for the classification of North Atlantic right whales upcalls. The Journal of the Acoustical Society of America, 149(4), 2520-2530. [106]

- Fabio Frazao, Oliver Kirsebom, Bruno Padovese, and Stan Matwin., Embedded Deep Learning for Underwater Acoustics, The Journal of Ocean Technology 15, No. 3, 174-175 (2020)

1.1.2 Conference Presentation

- Machine Learning in Marine Bioacoustics. Sobey Fund for Oceans - Sustainable Ocean Conference 2021, Dalhousie University, Halifax, NS, September 2021.

- Adaptable, Open-source Deep Learning NARW Vocalization Detection Tool - North Atlantic Right Whale Consortium, New Bedford Whaling Museum, New Bedford, Massachusetts, October 2022.

1.1.3 Workshop Report

- Fabio Frazao, Bruno Padovese, and Oliver S Kirsebom. Workshop report: Detection and classification in marine bioacoustics with deep learning. arXiv preprint arXiv:2002.08249, 2020.

1.1.4 Invited Talks

- Data Augmentation: Improving your Datasets. MERIDIAN Winter Webinar Series: Sound detection/classification with deep learning, Halifax, NS, Nov 2020.

- Methods to Overcome Data Scarcity. Detection and Classification in Marine Bioacoustics with Deep Learning, MERIDIAN and Ocean Networks Canada Workshop, Victoria, BC, Nov 2019.

- Augmentation Unchained Project. Detecting Whale Calls with Deep Neural Networks, MERIDIAN-DAMMA workshop, Quebec City, QC, Sep 2019.

1.2 Outline

The remainder of the report is organized as follows:

In Chapter 2, we present a bibliographic review of existing works on ML and DL approaches for the detection and classification of marine mammal vocalizations. In addition, we explore studies that employ ML techniques to combat the limited availability of annotated PAM data to train effective DL-based detectors and classifiers in underwater bioacoustics.

In Chapter 3, we introduce the marine mammal species that are the focus of this study and describe the common denominators across all phases, including the datasets used, the spectrogram computation process, the neural network architecture, and the performance metrics employed to evaluate the effectiveness of each approach. This chapter provides the foundational elements upon which subsequent chapters will build from.

Chapter 4 presents a data augmentation approach for improving the performance of DL-based detectors and classifiers. We introduces the study-case, PAM data, and data preparation steps used in the study. Next, we present the neural network architecture and data augmentation algorithms used to train the models. Finally, we describe the experimental steps taken and discuss the experimental results of the approach.

Chapter 5 explores a TL approach in order to improve the performance of a DNN that was originally trained on vocalizations from one environment to different environments. We present the PAM datasets from different sources used in the study and the data preparation steps. Then, we discuss and explore different TL strategies for PAM data and discuss the experimental results.

Chapter 6 presents a synthetic signal generation approach to address data scarcity in training DNNs on datasets where no annotations are available. We outline the methodology for creating realistic synthetic marine mammal vocalizations. We analyse which steps of the synthetic signal generation process contribute more to training a DNN to detect real vocalizations. Finally, we assess the effectiveness of this approach on a dataset with no annotated data and discuss limitations.

Finally, in Chapter 7 we summarize our findings, discuss possible limitations of this research, outline promising new lines of research and indicate potential future

work.

Chapter 2

Background and Related Works

Applying automated detectors to identify marine species and distinguish between vocalization types, is already a well established practice. Several PAM detection and classification systems employ a binary classifier that is responsible for verifying the occurrence of a signal of interest (detection) and a binary or multi-class classifier that determines the source of the signal (classification) [136]. For instance Kirsebom *et al.* [65] uses a sliding window of $3s$ and step size of $0.5s$ to compute a series of classification scores between $0-1$ and then average these scores over a $2.5s$ window to verify the presence of NARWs, significantly reducing the presence of false positives. In [132], a set of spectral and temporal criteria were used to validate captured echolocation clicks of beaked whales from a detection algorithm. In all cases, however, a final manual validation of the detections is common, reinforcing the need for robust methodologies. With respect to the classifiers involved in detecting marine mammal vocalizations, existing approaches can be grouped into two different classes, those that involve a set of extracted features that are then fed to a classifier, and more recent approaches involving DL. In Section 2.1 we review a couple of traditional detectors and classifiers, and in Section 2.2, we present DL-based studies for PAM and techniques to combat data scarcity.

2.1 Traditional Approaches

Initially, detection and classification of marine mammal vocalizations has been supported by signal processing techniques or classical ML methods such as shallow neural networks [105], random forests [46], support vector machines [36], classification and regression trees [100], Gaussian mixture models [117] and contour-based techniques [123]. In general, these methods work on a set of features that have been extracted from the data rather than on the raw audio itself. These features can have a high impact on the models performance, and therefore need to be carefully considered

when building a classifier. The main disadvantage of this approach is that the process of manually choosing which features to pass to a classifier and computing them from the raw audio data can be highly application specific, and may involve complex computational and mathematical steps that requires expertise in multiple domains [8, 13, 6, 105].

Works that employ classical ML algorithms have been proposed for the detection and classification of a number of marine species. Pourhomayoun *et al.* [113] used 18 handcrafted features such as event duration, signal-to-noise ratio (SNR), average bandwith and center frequency as input to a shallow neural network to classify bioacoustic signals. In [6], two specific sets of features based on short-time Fourier transform (STFT) and wavelet packet transform were fed to a shallow neural network to classify blue whale calls. Both methods leveraged the low frequency characteristics of a blue whale call to compute the feature sets only on two low frequency subbands. Ibrahim *et al.* [55] employed Mel Frequency Cepstral Coefficients (MFCC) and Discrete Wavelet Transform (DWTs) to extract features of NARW upcalls. These features were then used by a Support Vector Machine (SVM) and the K-nearest neighbors (KNN) algorithm for upcall detection.

Among the signal processing techniques, many algorithms have exploited the time-frequency structure of marine mammal vocalizations. Gillespie [43] developed an algorithm to distinguish right whale calls from other animals and anthropogenic sounds. The algorithm uses an edge detection algorithm and a gaussian kernel to extract 'outlines' of the signals on a spectrogram. Next, a number of features are computed such as duration, bandwidth and details of the frequency contour to be used by a classification function. In [38] transient signals that had similar time-frequency features to a particular pygmy blue whale call were searched by a signal recognition algorithm. The computed features included signal duration, the frequency band and the slope of frequency change with time. Another study [101] introduced a Minke whale detecion algorithm that searched for frequency features of 'boing' calls without having to first compute the continuous spectrogram, thereby reducing processing time. The algorithm employs a waveform envelope detector, a frequency characteristic matching on peak and side-band frequency features and the estimation of the duration of the boing to classify the sounds. In particular, the authors showed that the proposed method

drastically reduced false alarm rates of Minke whale calls originating from sounds produced by humpback whales. Simard *et al.* [130] applied a NARW upcall detector based on spectrogram cross-coincidence with a synthetic upsweep call template that was originally proposed in [95] for blue and fin whales.

2.2 Deep Learning

In contrast to traditional detection and classification algorithms, DNNs, the main component in DL-based systems, introduces the concept of representation learning through which much of the feature extraction process is incorporated into the neural network architecture. In representation learning, the multiple layers of the DNN will automatically learn to extract simpler features and aggregate them into progressively more complex non-linear representation of the data, thereby replacing the traditional feature extraction steps. In theory, the shallow layers of the DNN will learn simple, more generic features while the deep layers will learn more complex, specific features for a given task. Given enough data, DL-base models can outperform traditional ML approaches as the features learned by the DNN can often be more useful for a given task than manual feature engineering. In addition, DL-based systems can be easily adapted to other settings, because the same DNN can be trained to perform a wide range of tasks and its response can be improved or adjusted by simply feeding the network new data [104]. In practice however, most DL-based detectors and classifiers still use some sort of pre-processing step such as re-sampling, denoising, or the computation of spectrograms [139, 127, 65], but such pre-processing steps are often considerably simpler than heavy-feature engineering.

Advances in DNNs and DL techniques have shown substantial progress in many discriminative and generative tasks in several fields, for instance, image [49] and video [143] classification and object detection [148], face recognition [122], image to image translation [22], text translation [1] and generation [15]. DL has also been driving unprecedented developments accross a wide range of acoustic applications such as speech recognition [115, 108], speech and music synthesis [71], audio tagging [69], timbre transfer [53], among many other tasks [91, 97]. However, the applications of similar techniques to PAM is still relatively underused.

In terrestrial bioacoustic applications, deep learning (DL) has seen varied use.

Nolasco *et al.* [98] underscored challenges in detecting diverse animal sounds and the constraint of needing extensive species-specific annotated datasets for training DNNs. They introduced a few-shot learning framework for sound detection, employing a single model trained to recognize multiple sound events from just 5 examples each. To advance this method, the authors launched a public challenge. After three iterations, top methods combined meta-learning and transfer learning, emphasizing event duration and time resolution.

In another study, Ghani *et al.* [41] explore the utility of feature embeddings - vectors from machine learning models - for few-shot transfer learning. These embeddings can differentiate subtle acoustic details, enabling transfer learning between species and more accurate classifications. By leveraging pretrained embeddings from large-scale acoustic classifiers, the study proposes a method for species-agnostic classification across different taxonomic groups. Interestingly, the study found that models pretrained on extensive bird vocalizations provided better performance than those trained on data from bats, amphibians, or marine mammals. This suggests that certain feature embeddings might possess a stronger capacity for generalization across diverse bioacoustic tasks.

In [61], Kahl *et al.* developed BirdNET, a deep neural network designed for identifying vocalizations of 984 bird species from North America and Europe. Utilizing a task-specific architecture inspired by residual networks, the model comprises 157 layers and over 27 million parameters. Among several techniques employed to enhance its efficacy, domain-specific data augmentation was highlighted as an essential step which ensured robustness against ambient noise and overlapping vocalizations. BirdNET's was shown to perform comparably with more complex architectures used in areas like image object detection. By emphasizing the importance of high temporal resolution in input spectrograms, the model achieved high precision in various test scenarios. Specifically, BirdNET attained a mean average precision of 0.791 for single-species recordings and demonstrated consistent correlations with expert observations in eBird (a citizen science platform for submitting observation of birds) hotspots.

In underwater bioacoustics, Zhong *et al.* [149] employed an ensemble convolutional neural network (CNN) – a type of DNN – trained on spectrograms to detect beluga whale calls and whistles. In order to create an ensemble model, the authors took the

weighted average prediction of four different models based on commonly used DNN architectures. Kohlsdorf *et al.* [68] trained an autoencoder composed of convolutional and recurrent layers on spectrograms to perform several tasks related to dolphin vocalizations such as signal detection, signal type classification between background noise, echolocation clicks, bursts, and whistles, as well as KNN clustering based on signal type. Shiu *et al.* [127] trained a variety of DNNs to detect NARW upcalls and compared the results to traditional methods. Similarly, Kirsebom *et al.* [65] trained a ResNet [49], another type of DNN, to detect NARW upcalls, but they focused on a different geographic location and provided insights into the importance of dataset size and variance for training DNNs.

A number of studies have trained DNN to identify odontocetes clicks. Luo *et al.* [82] presented a method based on a one-dimension CNN to distinguish between click and non-click signals. The raw audio signal was used as input for the network and was first subjected to a high-pass filter pre-processing step. In a following study [145], the authors expanded the approach to automatically classify echolocation clicks from different species of odontocetes. Another study [11] developed two distinct sperm whale echolocation click detectors based on a CNN and recurrent neural networks (RNN). The CNN was used to classify spectrograms while the RNN was employed for a wider variety of classification tasks such as vocal clan classification and individual whale identification. Jiang *et al.* [59] expands the application of CNNs to odontocetes vocalizations to detect and classify whistles produced by killer whales and long-finned pilot whales using denoised spectrograms.

While DL-based approaches have shown great promise as versatile detectors and classifiers so far, many of these algorithms rely on large amounts of annotate data to accurately detect and classify marine mammal vocalizations. When this is not available, as is often the case in underwater bioacoustics, it is reasonable to employ ML strategies such as data augmentation and TL to mitigate the effects of data scarcity. In Section 2.2.1 we presents existing data augmentation research in PAM while 2.2.2 summarises TL research for PAM.

2.2.1 Data Augmentation

Simple data augmentation procedures have been used in many works as a relatively easy approach to gaining model robustness. Techniques such as time shifting, pitch shifting and noise injection are common practices often employed by researchers when developing detectors and classifiers. Some of these techniques are applied directly to the time-series data, while others are applied to the spectrogram representation, and some can be applied to both. Time shifting the spectrogram view in either direction to generate additional examples was employed by Shiu *et al.* [127] for NARW detection. Li *et al.* [73] applied several data augmentation techniques directly to the time-series rather than the spectrogram. The techniques included time and pitch shifting, time stretching, adding background noise and volume control (amplifying or attenuating the original audio) to generate more data. Spectrograms of all audio samples were then computed to train a CNN to detected and classify the dolphin whistles. Similarly, Mishachandar and Vairamuthu [94] injected artificial noise to samples in order generate additional samples along with the time and pitch shifting transformations for ocean noise classification. Waddell *et al.* [141] also employed a time shift augmentation approach to train a CNN for the classification of fish calls. Specifically, the authors shifted sepctrograms in time by small random amounts. Vickers *et al.* [139] used data augmentation to add several types of synthetic noise representative of the testing environment to improve a NARW classifier. In another study [10], an augmentation procedure involving two steps was proposed. In the first step, intensity, pitch, and time augmentation were conducted while in the second step noise injection was carried out. A CNN was then trained as a killer whale detector and classifier.

Another domain of data augmentation that addresses challenges arising from limited and imbalanced datasets is synthetic data generation. Computer-generated synthetic signals have been extensively utilized in marine mammal communication research, offering significant advantages over natural signals collected in sea trials. These advantages include not only data augmentation for training classifiers but also providing a controlled setting for conducting experiments and testing focused hypotheses [134]. Synthetic signals have been employed to investigate animal behavior, such as social communication [63, 3], and female mate choice [3, 114]. In other studies

[12, 30], synthetic signals were used to explore how distortion caused by signal transmission through the underwater acoustic environment affects automatic detection and classification methods, whose performance is highly dependent on the acoustic properties of the environment [95, 51, 129]. Binder [12] generated synthetic calls of bowhead and humpback whales to obtain clean signals free from acoustic propagation effects. In another example, Mercado *et al.* [92] trained a neural network to classify the propagation distance of recorded sounds, aiming to measure whether humpback whales used frequency degradation to estimate the range of singing whales. Although these synthetic signal generation approaches were not specifically intended for dataset augmentation, they could be similarly applied for generating additional data for training DNNs.

In the context of data augmentation, researchers have explored various approaches to generate synthetic PAM data for detecting and classifying marine mammal vocalizations. These methods range from using images containing unrelated but similarly shaped patterns [75], to creating new vocalizations through DL-based generative methods [76], and augmenting signals by propagating them through new environments. In Li *et al.* [75], synthetic data resembling delphinid whistles were created by injecting primitive shapes into spectrogram patches of whistle-absent recordings. These primitive shapes were derived from contours in the computer vision domain, such as images of buildings and people. The synthetic data was then used to augment a PAM dataset for training a DNN to extract dolphin whistle contours. In a subsequent work, Li *et al.* [74] introduces a novel approach of training models using pseudo-labels, approximate whistle labels generated using previously established whistle extractors that necessitate minimal human-annotated data. Training deep neural networks with these pseudo-labels presents challenges due to their inherent inaccuracies compared to human-generated annotations. The authors propose an improved loss function to compensate the inaccuracies created by the pseudo-labels. In another study, Duc [30] augmented a blue whale dataset by propagating randomly extracted samples through a new acoustic environment, effectively creating new whale calls subjected to different distortions.

The emergence of GANs has provided a powerful framework for creating realistic speech samples, enabling the generation of more realistic synthetic signals. Li *et al.*

[76], expanded their data augmentation procedure for generating delphinid whistles by employing GANs to generate synthetic data. Similarly, Shah [125] relied on DNNs based on GANs to generate synthetic Odontoceti whistles. While these DL-based generative methods have the potential to create highly realistic samples, they also pose significant limitations. First, being DL-methods, these generative networks require large quantities of training data, which limits their application in cases where only small amounts of annotated data are available or no annotated data exists. Second, GANs-generated samples can contain artifacts or have missing regions. Third, it is often difficult fine-tune generations to address a specific focus. This can lead to a GANs-based model trained to generate whale calls producing samples with no vocalizations or faint vocalizations. Consequently, a sample selection evaluation is sometimes required to validate the generated samples. However, despite these limitations, GANs remain a promising avenue for the generation of synthetic data in the field of underwater acoustics. Even though their application in underwater acoustics is still in its infancy, continued research and advancements in GANs can lead to significant improvements in marine mammal detection and classification models.

2.2.2 Transfer Learning

The most widespread TL technique for PAM is pretraining, whereby a DNN that has already been trained for another task is used to initialize the new DNN training process for a similar task. TL from image datasets such as ImageNet [27] has been a common practice in PAM research, and while quite different from the data in bioacoustics, it has nonetheless shown significant improvements to the final performance of detectors and classifiers. ImageNet is a large dataset containing annotated photographs for developing computer vision algorithms and often serves as a benchmark dataset in many studies. The principal argument for this approach is that the shallow layers of the DNN trained on ImageNet will learn low-level generic features such as basic shapes that can be shared to a wide range of different tasks, including bioacoustics. A pretrained network on the ImageNet dataset was used to detect and classify killer whale, long-finned pilot whale, and harp seal vocalizations in [81], showing that significantly less data was needed to achieve a satisfactory performance. The same TL approach was used in [149] to train three out of four models used in ensemble

learning for the detection of beluga whales and in [72] and [28] for bird species and bird song classification respectively.

More recently, some studies have begun using Google's AudioSet [39] and VGG-Sound [20] (diverse datasets containing audio from YouTube videos) to pre-train a DNN. In [23] transfer learning from AudioSet was used to develop a DL-system to tag arctic ecoacoustic recordings while [7] pre-trained a CNN on the VGG-Sound set to detect two audio-visually distinctive actions in wild chimpanzees. In a related study that also employed TL to address the issue of insufficient annotated data in PAM, Dufourq et al. [31] explored the use of 12 pre-trained DNN architectures on the ImageNet dataset, which is a vast collection of images, as the source models for TL. These pre-trained models were then adapted to classify vocalizations of four terrestrial endangered species, achieving satisfactory performance with as few as 25 annotated samples for each species.

Another approach used in TL is to make some layers of the pretrained DNN as untrainable (freeze), such that the information learned by those layers is conserved. Bermant *et al.* [11] froze the feature extraction layers of a RNN and replaced the classification layer. The model was then retrained and tested on unseen data to classify sperm whale echolocation clicks. Thomas *et al.* [136] trained a VGG-19 network (another type of CNN), to classify low frequency vocalizations from different species of marine mammals. All sixteen convolutional layers (the feature extraction layers) are then frozen and the model is trained only on the last three layers to adapt to humpback whale sounds.

Chapter 3

DNNs for the Detetion and Classification of NARW Upcalls

The typical process for developing DL-based detectors and classifiers often follows several key common steps. In this chapter, we establish the common foundational elements upon which we will build and evaluate the methodologies and strategies explored in the subsequent chapters. We will present the marine mammal species and vocalization of interest, the chosen datasets, the selected audio representation, the neural network architecture, and the evaluation metrics. By providing this groundwork, we ensure a cohesive and comprehensive understanding of each phase of the research and their comparative effectiveness and applications in detecting and classifying marine mammal vocalizations. We note, however, that while most of the elements presented in this chapter are consistent across all phases of this research, some elements may differ. Each chapter contains a description of the specific elements used and any additions or deviations from the common elements are defined as necessary.

3.1 North Atlantic right whale

This thesis focuses on developing a DNN model for the acoustic detection of NARW from PAM data. The NARW is a large baleen whale species that is designated "Critically Endangered" by the International Union for Conservation of Nature (IUCN), due to its small and declining population [24]. By the end of 2021, only 340 ($\pm$ 7) NARW were estimated to remain [112]. The species' decline is primarily driven by anthropogenic activities, with ship strikes and entanglement in fishing gear being major causes of mortality [66, 126]. Sub-lethal stressors such as ship noise [118] may also impede the species' recovery. In light of these issues, the NARW has garnered major research interest over the last several years in an effort to conserve the species. This includes the development of accurate acoustic detectors and classifiers, which would be invaluable for informing mitigation efforts such as management of shipping

and fishing activities [127].

In this thesis, we are concerned with detecting the "upcall" vocalization of the NARW. The NARW upcall is a stereotyped low-frequency upsweep (typical frequency range 100 – 300 Hz) that lasts approximately one second, and is frequently produced by both sexes under various behavioural contexts [110, 109]. These characteristics make the upcall an ideal candidate to identify NARWs acoustically, and thus it is often used in detectors and classifiers for this species (e.g., [130, 127, 65]). An online repository containing many examples of NARW upcalls and other vocalizations can be found in the Watkins Marine Mammal Sound Database [121] [1]. In the following chapters, we present several methods and strategies aimed at enhancing the performance of a DNN model when detecting NARW upcalls, particularly in the context of annotation scarcity.

3.2 Datasets

This thesis relies on three datasets containing NARW upcalls: (i) recordings collected in the Gulf of Maine (GOM) off the coast of Massachusetts, US, which were used for the Detection, Classification, Localization and Density Estimation (DCLDE) 2013 workshop challenge [44]; (ii) recordings collected in the Gulf of St. Lawrence (GSL) by [130]; (iii) and recordings collected in the Emerald Basin (EMB) off the coast of Nova Scotia, Canada by the Department of Fisheries and Oceans Canada (DFO) [32]. The specific geolocations for each deployment are illustrated in Figure 3.1. The GOM[2], GSL[3] and EMB[4] datasets have all been made publicly available. It is important to note, however, that the NARW's habitat encompasses and extends beyond the Gulf of Maine, the Gulf of St. Lawrence, and the Emerald Basin, where recordings were collected for this study.

The GOM dataset was collected during one-week long deployments in 2000 and 2009, with an array of bottom-mounted Marine Autonomous Recording Units (MARUs; [18]) developed by the Conservation Center for Bioacoustics at Cornell University[5].

[1] The NARW examples can be found at https://whoicf2.whoi.edu/science/B/whalesounds/bestOf.cfm?code=AA3A

[2] The GOM data is available at https://soi.st-andrews.ac.uk/dclde2013/

[3] The GSL data is available at https://doi.org/10.20383/101.0241

[4] The EMB data can be found at https://doi.org/10.3389/fmars.2022.976044

[5] Conservation Center for Bioacoustics; https://www.birds.cornell.edu/ccb/technology/

Figure 3.1: Map illustrating the deployment locations for the GSL (Top Figure) GOM, and EMB (Bottom Figure) datasets. Data was collected using varying hardware and over different time periods, as detailed in Section 3.1. The location indicated for the GOM deployment is approximate, due to the unavailability of specific coordinates.

Data from the 2009 deployment, collected between March 28 and April 3, contains NARW upcalls. In contrast, the data from the 2000 deployment, collected between May 19 and May 25, does not include any identified NARW upcalls. The MARUs were deployed approximately 3 meters above the seafloor with no surface expression. All audio files in the GOM dataset are 15-minutes long with a sample rate of 2 kHz and were recorded continuously (i.e., they span the entire week without gaps or interruptions). The dataset comprises data from all 7 days of recording from one recorder in both recording years. In total, the dataset contains 336 hours of recordings. All audio files were subjected to a full manual analysis by an expert and all NARW upcalls

were annotated.

The GSL data were collected between 2015 and 2019 at six different stations across the GSL through surface deployments tethered to a buoy and bottom deployments moored approximately 5 m above the seafloor. Each bottom recording station was equipped with an AURAL-M2 autonomous underwater recorder[6] while the surface deployments were equipped with an ic-Listen hydrophone[7] connected to a real-time ocean observing Viking buoy[8].

A subset of the GSL recordings were downsampled to 1 kHz and processed using a time-frequency-based detection algorithm (TFBD) to detect candidate NARW upcalls, which were then validated by an expert. The algorithm follows [90] and searches spectrograms of the recordings for NARW upcalls based on spectrogram cross-coincidence with a synthetic upsweep call template originally proposed in [95] for blue and fin whales. The upsweep call template used for NARW upcalls had a bandwidth of 20 Hz, was 1-s long, and had a frequency sweep kernel between 100 to 200 Hz [130]. All validated detections had the NARW upcall time centered based on the midpoint of the upcall determined by the TFBD algorithm with minor temporal misalignment of up to 0.5 s [65]. In this manner, a large number of 3-second long audio clips were extracted from the data, collected between June 9, 2018, to July 17, 2018, and between May 31, 2019, to August 21, 2019. These are dubbed as GSL_A for the clips originating from the surface deployments, and GSL_B for the clips from the bottom deployments. Additionally, a smaller subset of the GSL recordings from the bottom deployments were subjected to a full manual analysis by an expert. This effort resulted in 50 audio files collected between August 17, 2015 and January 10, 2017, each 30 minutes in duration, amounting to a total of 25 hours of recordings sampled at 32 kHz, in which every NARW upcall has been annotated (this subset is named as GSL_B*). Both the 3-second clips and the 30-minute continuous files form the the GSL dataset used in the present work. Further information on the GSL dataset data collection and specific deployment locations can be found in [65].

The EMB data were collected using a bottom-mounted Autonomous Multichannel Acoustic Recorder (AMAR; JASCO Applied Sciences, Ltd.) deployed from May

[6]Multi-Electronique Inc.; http://www.multi-electronique.com/aural.html
[7]Ocean Sonics; https://oceansonics.com/product-types/iclisten-smarthydrophones/
[8]Multi-Electronique Inc.; http://www.multi-electronique.com/buoy.html

24, 2015 to April 20, 2016 (EMB_2015) and from September 15, 2016 to November 30, 2017 (EMB_2016) in the Emerald Basin. The acoustic recorders were placed approximately 20 m above the seafloor. The recording schedule was duty cycled and alternated between two sampling rates, such that data were sampled at 8 kHz for 11 min 18 s, followed by 250 kHz for 1 min 4 s. This cycle was repeated every 20 min. For this study, only the 8 kHz recordings were used. In total, the EMB_2015 dataset contains $\sim 7,247$ hours of recordings, and the EMB_2016 dataset contains $\sim 9,608$ hours of recordings. All recordings were processed with the Low Frequency Detection and Classification System (LFDCS; [8]), using the call library from [8]. All calls that were classified as a NARW upcall call type by LFDCS with a Mahalanobis distance $<= 3$ were reviewed and validated by an expert, following the protocol described in [32].

Table 3.2 displays the total number of annotated NARW upcalls in each dataset. Each dataset has its own unique characteristics, advantages and limitations. For instance, whilst the GOM dataset has the largest amount of annotated upcalls for training, its data was collected over only one week of monitoring, which may not provide an accurate representation of the environmental soundscape during other seasons. On the other hand, the EMB dataset was collected over a longer stretch of time, thereby capturing more variability in the acoustic environment and NARW vocalization patterns, but only contains 479 annotated upcalls, a very low quantity for training a DNN. By using multiple datasets in this study, we aim to address each limitation through our methodology, ultimately creating more robust and generalizable DNN models for detecting and classifying NARW upcalls across diverse acoustic environments and time periods. We note that each phase of this thesis utilizes a different combination of these datasets. In each respective phase, we clearly specify which datasets are employed to ensure a thorough understanding of the methodology and results.

3.3 Audio Representation

The majority, but not all, of studies in the acoustic domain use a spectrogram representation of the audio data rather than the raw waveform. The main motivation behind using a spectrogram representation is that this is how researchers analyse

Dataset	Upcalls
GOM	**9,063**
GSL	**4,331**
GSL_A	800
GSL_B	2,374
GSL_B*	1,157
EMB	**479**
EMB_2015	232
EMB_2016	247

Table 3.1: Total number of annotated NARW upcalls in each dataset. The GSL dataset displays the total number of annotated upcalls contained in the GSL_A, GSL_B and GSL_B*. Similarly, the EMB dataset displays the total number of annotated upcalls contained in the EMB_2015 and EMB_2016 datasets. Training, validation and testing splits are defined in each chapter as necessary.

this type of data. One advantage of such an approach is that it is much faster to view a spectrogram than listen to potentially long recordings [136]. Spectrograms visually represent the frequency content of a signal over time, allowing for the easier identification of specific features and patterns, such as harmonics, as well as enabling more effective noise filtering within the frequency domain. Another advantage is that you can see sounds that are too low or too high in frequency to be audible by a human listener. Furthermore, by modelling the problem as an image task, it is possible to take advantage of powerful DNNs developed for image data. For instance, in speech recognition and speech synthesis tasks, it is common to pair mel spectrograms with DNNs [108, 71]. This type of spectral representation of the sound information resembles that of the human ear and can be useful for solving speech related tasks.

In light of this, current state-of-the-art NARW upcall detectors use magnitude spectrograms as input instead of acting directly on the raw audio data. This is true for conventional detectors [8, 130] as well as the more recent generation of deep-learning detectors [127, 65]. In this work, magnitude spectrograms with a frequency range of 0 Hz - 500 Hz are employed as input to a DNN to classify 3-second audio segments as containing a NARW upcall or not.

To produce the training sets for each phase of the reasearch, audio segments were extracted from each dataset as determined by the annotations. These segments were

designated as "positive" if they contained a NARW upcall and as "negative" otherwise. For the negative samples, the GSL clips and EMB_2015 dataset contained "negative" annotations that were false detections (hard negatives) from the TFBD and LFDCS algorithms, respectively. For the GOM dataset and the GSL 30-minute audio files, the negative samples were extracted by randomly isolating 3-second segments that did not overlap with the annotated upcalls. This process was also repeated to extract additional negative samples from the EMB_2015 dataset in order to complement the hard negatives with a more representative view of the EMB soundscape. The number of negative samples extracted from each dataset varies for each phase of the study, reflecting the specific requirements and methodologies employed. These variations are thoroughly detailed and explained in the corresponding chapters.

To compute the spectrograms, audio files from the GOM, EMB and GSL 30-minute datasets were downsampled to a sampling rate of 1 kHz to match the sampling rate of the GSL 3-second clips. Next, for all 3-second segments, the spectrogram representation was computed with a Hamming window size of 0.256 s (NFFT of 256 samples) and step size of 0.032 s, which represents an 87.5% overlap, and produces a spectrogram (2-d array) with time $\times$ frequency dimensions of 94 $\times$ 129. These settings are ideal for the detection and classification of upcalls [40, 130]. Finally, the resulting 2-d array was normalized to have a mean of 0 and standard deviation of 1. Figure 3.2 displays a spectrogram of a stereotypical NARW upcall using the specified parameters.

3.4 Neural Network Architecture

In this thesis, we employ a residual network (ResNet) [49] architecture to classify NARW upcalls. ResNet, a type of convolutional neural network (CNN), has exhibited strong discriminative capabilities across a variety of classification tasks and is relatively straightforward to implement. These attributes make ResNets well-suited for our study, as the primary objective is to investigate the potential of ML techniques to address data scarcity rather than to focus on the architecture itself. As with traditional CNNs, they employ layers of convolutional filters that transform the raw input data - in our case, a spectrogram - into useful features which are then fed to a traditional feed-forward neural network for classification.

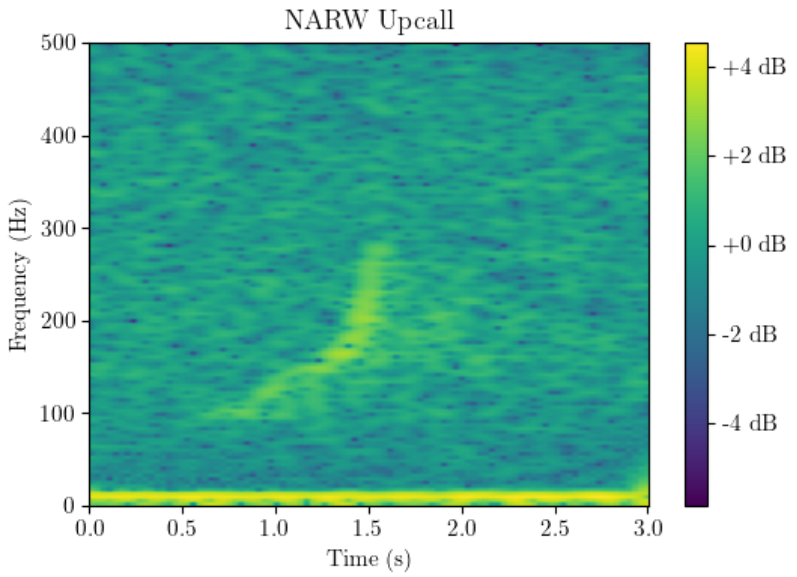

Figure 3.2: Spectrogram of a stereotypical NARW upcall, generated using the spectrogram computation parameters described in Section 3.3. The spectrogram illustrates the distinctive low frequency upsweep characteristic of NARW upcalls.

These features are implicitly learned during the training process by leveraging patterns in the time-frequency bins, producing weighted combinations of nearby elements which then pass through a non-linear activation function and serve as input for the next layer. In theory, the earlier, shallow layers learn low-level generic features while delegating the learning of specific features to the deeper layers of the DNN. However, very deep CNNs often suffer from a loss of generalization capabilities due to the vanishing/exploding gradient problem [49]. To help alleviate this issue, and have networks that generalize better, ResNets introduces simple "skip connections" in the network architecture that allow the learning procedure to bypass (skip) stacks of convolutional layers (a residual block), therefore preserving what was learned in the previous layers. This design improves generalization and enables training of deeper networks.

We utilize the widely adopted ResNet-18 architecture [49], a variant of the ResNet family, to classify the NARW upcalls. This particular architecure starts with an initial convolutional layer with 16 filters, followed by 8 ResNet blocks, each containing 2 convolutional layers with 3x3 kernels. The depth of the network increases by doubling the number of filters every 2 ResNet blocks. Each convolutional layer is accompanied

by batch normalization and employs rectified linear units (ReLU) [96] as the activation function. The residual blocks are then followed by a final batch normalization [57] layer and global average pooling [78]. At the end, a fully connected layer with a softmax function produces a score for each of the two possible classifications (upcall present or absent). These scores, ranging from 0 to 1, represent the network's confidence in each classification, and their sum equals 1. It is worth noting that there are other popular ResNet architectures (e.g., ResNet-34, ResNet-50, etc.) that could potentially be more powerful due to their increased depth of layers. However, given the limited amount of data, the fewer layers of ResNet-18, compared to its sibling architectures, allow for faster and more computationally efficient training without compromising performance. This characteristic is particularly advantageous when considering embedded PAM systems where power consumption and processing power are limited.

3.5 Performance Evaluation

For each phase of the experiments, a test set was reserved to evaluate the performance of the DNN model after training. This test set consisted of annotated data that was not used during the training process, allowing for an unbiased assessment of the model's ability to classify NARW upcalls. Specifics concerning which dataset was used for training and which dataset was reserved for testing, as well as any training and testing splits, can be found in the relevant chapters of this thesis. In these chapters, we detail the motivation for each experiment, discussing their objectives and providing a comprehensive description of the methodology employed.

To evaluate the network's performance and account for variability due to random initializations, all experiments were conducted five times using different initial network weights and biases. The results were then averaged to establish the mean. Samples were classified as 'positive' if the positive-class (NARW upcall detected) score computed by the model exceeded a certain threshold value, and were classified as 'negative' otherwise. Given a class score s and a detection threshold τ, the network's detection can be represented as follows:

$$D = \begin{cases} 1, & \text{if } s \geq \tau \\ 0, & \text{otherwise} \end{cases} \tag{3.1}$$

where D is the detection results, either positive or negative, $D = 1$ indicates a positive detection (NARW upcall) and $D = 0$ indicates a negative detection (no NARW upcall).

Model evaluation involved computing the precision, recall, and false positive rate (FPR) with respect to the upcall (positive) class. The precision measures the proportion of instances classified by the network as positive that indeed have positive labels (true positive), i.e., they were also labeled as positive by the expert annotator, as opposed to negative labels (false positive). Recall on the other hand, measures the proportion of positively labeled instances in the test set that were correctly classified as positive (true positive) by the network, as opposed to negative (false negative). Finally, the FPR identifies how many false positives are detected on average over a unit of time and can better reflect how the model performs in a real world application. In this work, we consider the FPR to be the number of FP per hour. The precision and recall can be computed as follows:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{3.2}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{3.3}$$

where TP is the number of true positives in the test set, FP is the number of false positives, and FN is the number of false negatives.

It is worth mentioned accuracy as an intuitive metric for evaluating performance. However, in the context of detecting animal vocalizations such as NARW upcalls, accuracy presents important limitations. This is primarily because it assumes equal risk and equal importance associated with false positives and false negatives [12], as well as equal importance between classes. Marine mammal vocalizations are generally sparse occurrences in vast amounts of continuous acoustic data, such as those collected by PAM systems. Consequently, a model that classifies the majority of the data as negative (i.e., no vocalization) may still achieve high accuracy despite failing to detect

the actual vocalizations. In such scenarios, metrics such as precision, recall, and FPR provide a more comprehensive understanding of a model's performance, as they account for the imbalance in class distribution and the potential costs associated with false detections and missed detections. For this reason, this thesis and many other works [65, 127, 75, 76] focused on detecting animal vocalizations favour evaluating a DNN model's performance using the previously described metrics of precision, recall, and FPR, over accuracy.

Chapter 4

Data Augmentation for PAM

The contents of this section were based on our work published in; Padovese, B., Frazao, F., Kirsebom, O. S., & Matwin, S. (2021). Data augmentation for the classification of North Atlantic right whales upcalls. The Journal of the Acoustical Society of America, 149(4), 2520-2530. [106]

ML models in general, and DL models in particular, greatly benefit from large amounts of labeled data, and in theory, the more labeled data available, the better these models should perform at their respective tasks. This notion is formalized within the Probably Approximately Correct (PAC) learning framework [137]; loosely speaking, a task is said to be PAC-learnable if by enlarging the training dataset we can increase the probability that the learning process produces a well-performing model, solving the task to within a prescribed level of accuracy. However, in practice, the availability of labeled data is always limited and it is often either unfeasible or expensive to acquire more labeled data. Therefore, data augmentation strategies that produce more data by generating them synthetically can be an inexpensive way to mitigate this problem.

While many different data augmentation strategies have been proposed, they all share the same goal of having DL models generalize better. By generating new samples through different techniques, an augmented dataset comprehends a more representative set of possible data points, thus minimizing the distance between what a DL model has seen during training and what it can see in a real world application. While other solutions for increasing generalization such as employing more complex architectures or TL exists, data augmentation addresses the core of the issue, that is, providing DL algorithms with more samples to train from [128]. This is done with the expectation that additional information can be extracted from the original dataset through augmentations. Another advantage of using data augmentation is that we can fine tune our approach to tackle a specific knowledge gap or deficiency

that our model has. For instance, if we know that the environment we want to apply our DL model to is close to a busy shipping lane, we can use this prior knowledge to transform our data to simulate a similar environment and make our model invariant against this type of noise.

Data augmentation techniques can range from simple "naive" transformations that do not take into consideration the problem being addressed to complex DL-based augmentations that are capable of modelling raw audio drawn from a training dataset. Naive augmentations could be characterized by their simplicity and include transformations based on signal processing functions such as pitch shifting [73], time stretching [73], time warping [108] adding noise [94], as well as transformation on spectrogram representations such as time shifting [127], and those inspired by computer vision such as random masking [108]. Data-based augmentations instead, leverage the user knowledge of the domain to transform samples based on the characteristics of the environment, and can include simple operations such as mixing signals [144] to sound propagation modelling to synthesise how a signal distorts as it propagates through the underwater environment [12]. Finally, DL-based [45] augmentations learn the distribution of the dataset feature space to generate new unseen samples drawn from this distribution.

One important consideration when using data augmentation techniques is label fidelity. Shorten and Khoshgoftaar [128] refers to this problem as the "safety of application", and consider an augmentation technique to be "safe" if there is a high likelihood that the label of the generated sample is preserved post-transformation. This problem is best described using the classic computer vision task of recognizing "6" and "9" or "b" and "d". In this case, transformations that flip the image horizontally or rotate it 180° would also transform the meaning of the labels and therefore are "unsafe". Generally speaking, knowing which transformation is "safe" to apply or not depends on the application and requires a certain level of domain knowledge about the data.

4.1 Dataset

In this phase, we have opted to work only with a subset of the clips data as we are interested in studying the effects of data augmentation on PAM data and not

on producing the best possible NARW classifier. For this reason, we selected the GSL_B clips dataset. We recall that this particular subset data were collected through bottom deployments moored approximately 5 m above the sea floor at four different stations, each equipped with an AURAL-M2 autonomous underwater recorder, and that the clips were extracted from longer recordings using a TFBD algorithm. For more information, refer to Section 3.2. In total, the chosen subset contains 3,888 3-s long sound clips, of which 1,514 were negative samples (with no right whale upcall) and 2,374 were positive samples (containing a NARW upcall).

For composing the training and testing sets, the data were split based on the temporal placement of each 3-s clips within the longer recording. A time, t_0, was defined to split the dataset in an 85:15 ratio between training/validation and testing, with samples extracted from times $t < t_0$ selected for the training set and samples extracted from times $t > t_0$ chosen for the testing set [65]. Following this procedure, the training/validation set consists of 3,309 clips, including 2,033 positive samples and 1,276 negative samples, while the testing set comprises 579 samples, with 341 positive and 238 negative samples. Table 4.1 summarizes the composition of the training/validation and testing sets, providing the number of positive and negative samples for each set.

	Training + Testing	**Training**	**Testing**
Total	3,888	3,309	579
Positives	2,374	2,033	341
Negatives	1,514	1,276	238

Table 4.1: Composition of the training/validation and testing sets for the GSL_B clips dataset. The table displays the number of positive (containing NARW upcalls) and negative (no NARW upcall) samples in each set, as well as the total number of samples.

4.2 Data Preparation

For the methodology described in this chapter, we adhered to the same spectrogram computational steps outlined in Section 3.3. Since the data used in this phase was already segmented into 3-seconds clips, contained negatives, and were already at required sampling rate of 1,000 Hz, no additional pre-processing steps were required.

4.3 Data Augmentation

In this phase, two augmentation methods, SpecAugment and Mixup that were originally introduced for Automatic Speech Recognition (ASR) and computer vision tasks respectively were studied and adapted to PAM data. Both methods have shown to significantly improve the performance of classifiers in their respective fields, and have been chosen for their performance boost potential and ease of integration in a ML pipeline. In the following, we describe these methods and explain how we adapted them to a PAM task.

4.3.1 SpecAugment

SpecAugment [108], as implied by its name, is an augmentation technique specifically tailored to dealing with spectrograms in acoustic discriminative tasks. Its primary purpose is to make an ASR model more robust against small interferences such as deformation and loss of bins in the time or frequency dimension. This is done by introducing three transformations to the spectrogram, namely time warping, frequency masking and time masking as shown in Algorithm 1. Given a spectrogram with τ time bins and ν frequency bins:

1. Time warping consists of a deformation of the spectrogram in the horizontal (time) direction, in which the corners of the image and the mid-points of the vertical edges are used as fixed points, while a vertically centered point, randomly positioned in the horizontal (time) direction within $(W, \tau - W)$, is shifted by an amount 0–W where W is the user specified max warp distance parameter. We modified the tensorflow implementation of the tfa.image.sparse_image_warp method [108] so that it allows for the mid-points of the vertical edges to be used as fixed points without also fixing the mid-points of the horizontal edge.

2. Masking is done by selecting a number of blocks of time bins or frequency bins and replacing all values within the blocks with a mask value.

    - Frequency masking works by first randomly selecting f frequency bins to be masked from the interval $[0, F]$, where F is a user-specified parameter. Then, a random frequency bin along the vertical axis, f_0, is selected from

$[0, \nu - f)$. Finally, all pixel values in the frequency bin range $[f_0, f_0 + f]$ are replaced with a constant mask value.

- Time masking consists of first selecting t time bins to be masked from the interval $[0, T]$, where T is a user-specified parameter. Then, a random time bin along the horizontal axis, t_0, is selected from $[0, \tau - T)$. Finally, all pixel values in the time range $[t_0, t_0 + t]$ are replaced with a constant mask value.

The mask value can in principle be anything, but often the minimum, maximum or mean value of the masked area is used. Figure 4.1. displays a Spectrogram from the dataset before and after applying SpecAugment.

Algorithm 1 SpecAugment

Input: A spectrogram S, max warp distance W, max masked frequency bins F and max masked time bins T

Output: $\overline{S}$ - The SpecAugmented Spectrogram

$\tau \leftarrow$ GetTimeBins(S)
$\nu \leftarrow$ GetFrequencyBins(S)
if TIME WARPING **then**
    $vc \leftarrow \nu/2$ {Vertical center}
    $p \leftarrow$ RANDOM(W, $\tau - W$) {Random point in the horizontal line}
    $sa \leftarrow$ RANDOM(0, W) {Shift Amount}
    $\overline{S} \leftarrow$ ImageWarp(vc, p, sa) {Perform time warping}
end if
if FREQUENCY MASK **then**
    $f \leftarrow$ RANDOM(0, F)
    $f0 \leftarrow$ RANDOM(0, $\nu - f$)
    Set a value to $\overline{S}[:, f0 : f0 + f]$ {Masking}
end if
if TIME MASK **then**
    $t \leftarrow$ RANDOM(0, T)
    $t0 \leftarrow$ RANDOM(0, $\tau - t$)
    Set a value to $\overline{S}[t0 : t0 + t, :]$ {Masking}
end if
return $\overline{S}$

Multiple, possibly overlapping, masking operations may be carried out for a single input spectrogram and the number of masks should be chosen based on the type of

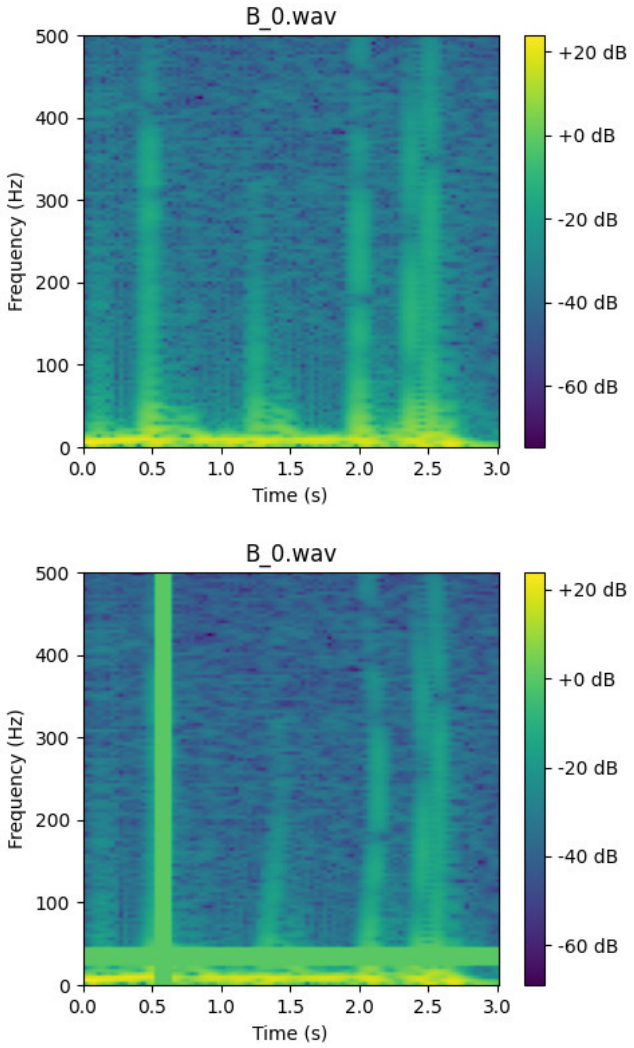

Figure 4.1: SpecAugment applied to a spectrogram. From top to bottom, the figures shows the the original spectrogram and the augmented spectrogram with time warping ($W = 8$), frequency masking ($F = 7$) and time masking ($T = 5$).

signal of interest and the characteristics of the spectrogram. Each of the three described transformations may also be disabled. For instance, according to the authors of Park *et al.* [108], time warping contributed only minor improvements in the overall performance of the ASR model. In this work, SpecAugment was applied to the Magnitude Spectrogram representation of the input audio, generating new augmented samples to feed to the DNN during training.

Being simple and computationally cheap, SpecAugment can be applied during

the training phase without significantly impacting training time. However, it is important to exercise caution when choosing the mask and warp parameters for each transformation, for instance how many frequency or time bins to mask. SpecAugment was originally proposed for ASR tasks, where the signals of interest are long and occupy most of the recording and masking large segments should not affect the overall structure of the signal, nor its label. However, in PAM detection and classification the opposite is frequently true, marine mammal vocalizations are often short and far and few in between long segments of silence or background noise. In this context, since the signals are short, it is possible to completely mask out the signal or distort it significantly if we choose large values for the parameters, thereby making the transformations non-label preserving. We contribute by adapting SpecAugment from an ASR context to a PAM detection and classification problem by proposing much smaller values for the masking and warping parameters but with more frequent occurrences of masking per recording.

In this study, even though SpecAugment provides transformations that may not resemble the kinds of interference that are common in hydrophone data, they may still be useful in terms of boosting model performance by generally increasing the diversity of the training data as we are working with substantially less amounts of data than used for training the ASR model in Park *et al.* [108]. In addition, by masking some time and frequency bins, the model will have to learn to predict the same outcome, using less available information. This results in a more robust representation of the data learned. Another important difference is that while Park *et al.* [108] worked with the Mel Spectrogram because it provides useful features for ASR tasks, we work with Magnitude Spectrograms.

4.3.2 Mixup

The Mixup [147] method generates a new training example, in our case a new waveform, by computing the linear superposition of two input arrays, including their labels. As it takes an array as input, it can work with both images [147] and raw audio waveforms [144].

Given two samples x_i and x_j from a training set and their associated labels y_i and y_j, a new virtual training example can be constructed by:

$$x = \alpha x_i + \beta x_j$$
$$y = \alpha y_i + \beta y_j \tag{4.1}$$

where $\alpha, \beta \in [0, 1]$ are the mixing ratio. Note that while the default algorithm uses $\beta = (1 - \alpha)$, our implementation gives a little more freedom by allowing a user defined α and β separately. The method is also described in Algorithm 2.

Algorithm 2 Mixup

Input: Two audio files x_i and x_j, their labels y_i and y_j and the α and β weights
Output: The new waveform x, and its label y

$x_i, y_i \leftarrow$ LoadAudio() {Load the first audio file}
$x_j, y_j \leftarrow$ LoadAudio() {Load the second audio file}
if β is null **then**
    $\beta \leftarrow (1 - \alpha)$, and $\alpha, \beta \in [0, 1]$
end if
$x \leftarrow \alpha x_i + \beta x_j$
$y \leftarrow \alpha y_i + \beta y_j$ {The labels can be represented as one-hot vectors}
return x, y

Mixup can be implemented with just a few lines of code and introduces minimal computational overhead, whilst incorporating the knowledge that linear interpolation of audio feature arrays will produce linearly interpolated associated labels [147]. This allows us to incorporate the method during training, by creating a virtual sample from two input arrays extracted from the training set. In practice, Mixup often leads to significant improvements in the models performance [147, 144].

In computer vision, Mixup or similar mixing techniques [147, 56], whereby two or more images are mixed together by averaging their pixel values, have shown to be surprisingly effective in boosting the performance of ML models at discriminative tasks, even as this may seem counter intuitive [128]. In most computer vision tasks, a pixel in an image always corresponds to a unique physical object, and it is usually safe to assume that adjacent pixels also correspond to the same object. Hence, two visual objects mixed together may result in an object that makes little sense to a human observer. In the context of audio data, however, the mixing not only helps the model learn more robust representations of each signal, but also results in realistic sounds that can be interpreted by a human listener in a straightforward and intuitive manner

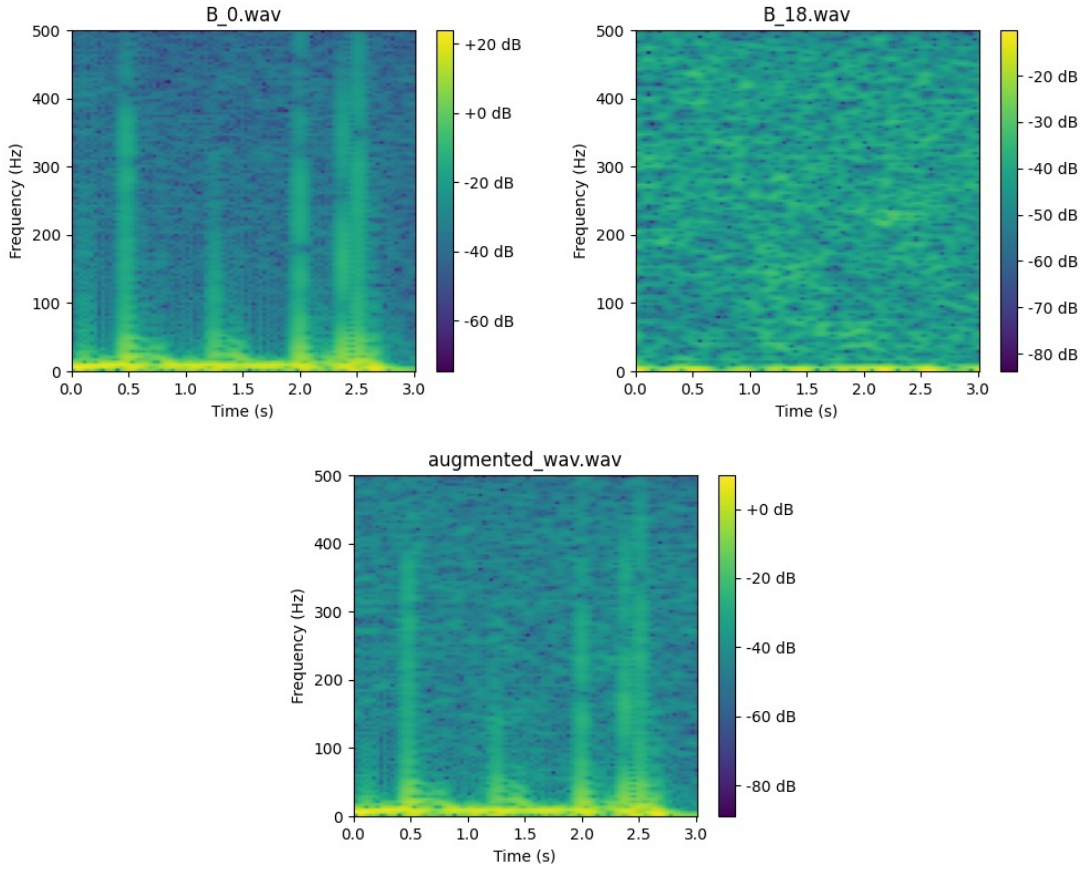

Figure 4.2: The Mixup method as applied to two audio samples. The top two images are the source samples which are mixed to become the augmented sample.

because sounds can and do overlap naturally. Sounds are mechanical pressure waves, and when two such waves interfere the resulting wave is the linear superposition of the two.

For the purpose of this work, the new mixed samples are generated before training and added to the training database, which permits additional pre-processing steps to be carried out such as the spectrogram computation. In addition, we hypothesise that the linear superposition of two input audio samples (Figure 4.2) alone while maintaining a single label for the new augmented sample is enough to achieve significant gains. Therefore, for all our experiments after applying Mixup, instead of preserving the linearly interpolated labels Eq. (4.1), we discarded one of the labels Eq. (4.2). This further simplifies implementation as we can now assume each clip belongs to a single class. Under this setting, as to not assign a negative label to a positive sample,

all new negative samples were augmented from two original negative samples. As to be consistent with the original dataset, which only has one upcall per 3 s clip, all new positive samples were augmented from one original negative sample and one original positive sample. Furthermore, to maintain consistency when generating a new positive, the α parameter will always be the weight for the negative sample, while β will always be the weight for the positive one. Finally, all mixed samples are then normalized to the interval $[-1, 1]$.

$$y = \begin{cases} 0, & \text{if } y_i = y_j = 0 \\ 1, & \text{otherwise} \end{cases} \tag{4.2}$$

The main difference between SpecAugment and Mixup in relation to their augmentation capability is that SpecAugment enhances the model's robustness against partial loss of information by providing samples with missing information. Additionally, SpecAugment poses a more challenging problem for the DNN, requiring it to produce the same output while relying on less information. Meanwhile Mixup enriches the training set by generating new audio that combines information from two recordings. This allows the model to learn a more robust representation of each class and can potentially strengthen it against indecision. Ultimately, both algorithms enhance a DNN model's performance by expanding the range of diverse data it is exposed to during the training process.

4.3.3 Experimental Protocol

Firstly, when augmenting the original dataset, for all experiments, we ensure that our binary classes will have the same number of positive and negative samples after augmentation. Hence, since the original dataset contains fewer negative samples than positive samples, all augmented datasets will contain exactly 757 more augmented negative samples than augmented positive samples.

The network was trained for $N = 100$ epochs and with a batch size of 128 samples per optimization procedure. All experiments were conducted using the ADAM optimization algorithm [64] with the default parameters of initial learning rate of 0.001, decay of 0.01, β_1 of 0.9, and β_2 of 0.999. Finally, all experiments were conducted

on an NVIDIA TITAN V GPU with 12 GB of memory. For determining the network's performance, we computed the mean recall and precision, as established in Section 3.5, on the test set for each experiment.

4.3.4 Full Dataset Protocol

For the purpose of creating a useful augmented set, we initially evaluated the best parameters for both methods. Starting with Mixup, we extensively experimented with different α and β values to assess their impact when creating new positive samples for the training set. Specifically, we evaluated if giving more emphasis to either of the original samples would help the DNN model learn the desired patterns better than a balanced approach ($\alpha = \beta$). For this purpose, we tested the following combinations of α and β values, $\alpha = 0.2$ and $\beta = 0.8$; $\alpha = 0.3$ and $\beta = 0.7$; $\alpha = 0.4$ and $\beta = 0.6$. Based on these investigations, only a modest effect was found when varying the weights in the given interval. Therefore, when applying Mixup for the remaidner of the experiments in this work, the α and β weights will always be set to 0.5 when augmenting data for either class.

For SpecAugment, we originally compared the effects of applying all three transformations at the same time (frequency masking, time masking, and time warping) versus applying the masking operations and time warping individually. We augmented the dataset to have 10,000 samples from each class. For the time warping operation, we set the warping parameter to $W = 8$, while for the masking operations, we applied six time and frequency masks to each augmented sample with frequency masking parameter $F = 7$ and time masking parameter $T = 5$. We found these parameters to be ideal when working with the data. Specifically, we found that using multiple smaller masks (masking fewer time or frequency bins) was more effective than fewer bigger masks. While some variation was found when isolating the three operations, this was not substantial, and for all other experiments involving SpecAugment, all three transformations will be used when augmenting data for either class.

Finally, with these initial observations we investigated the DNN model performance when training on progressively larger amounts of augmented data. Starting with the full original dataset, we augmented the training set to 10000 total samples, and then incrementally added 5000 augmented samples at each step up to 40000 total

samples, using either SpecAugment or Mixup.

4.3.5 Undersampled Dataset Protocol

The first step in understanding the effects of augmentation in a scenario of data scarcity was to draw a set of baseline performances when feeding the DNN only a portion of the original training set, without any sort of data augmentation. To this end, we assess the performance of the DNN when training on just 200 original samples (100 from each class), and incrementally adding another 200 original samples (100 from each class) at each new training procedure. Next, we investigate the effects of data augmentation by progressively increasing the amount of augmented data available for training a DNN. From the baselines established in the previous experiment, we increased the amount of augmented samples at each step by 2000 up to 8000.

4.4 Results

4.4.1 Full Dataset

We report the DNN model's performance when training the model on progressively larger amounts of augmented data in Figure 4.3. The dashed line represents the baseline performance when training with all the available original data without augmentation. We observe that the model greatly benefits from the added augmented samples, with recall improving from 89.0% up to 93.0% and precision improving from 85.9% up to 90.7% in the best scenarios. Furthermore, it is worth highlighting that the model already achieves high precision and recall rates with a relatively modest number of augmented samples. With 10000 total samples (5000 from each class), the model achieves close to 90% precision and recall with both Mixup and SpecAugment. By increasing the total samples to 20000 (10000 from each class), it is possible improve the performance even more while also achieving a lower standard deviation. With 20000 total samples, the model attained 90.7% precision and 90.8% recall with the Mixup method and 90.1% precision and 91.3% recall with SpecAugment. Finally, we observe that under our setup, it took $\approx$ 27 minutes to train the model without any augmented samples, increasing by $\approx$ 45 minutes for every additional 5000 augmented samples.

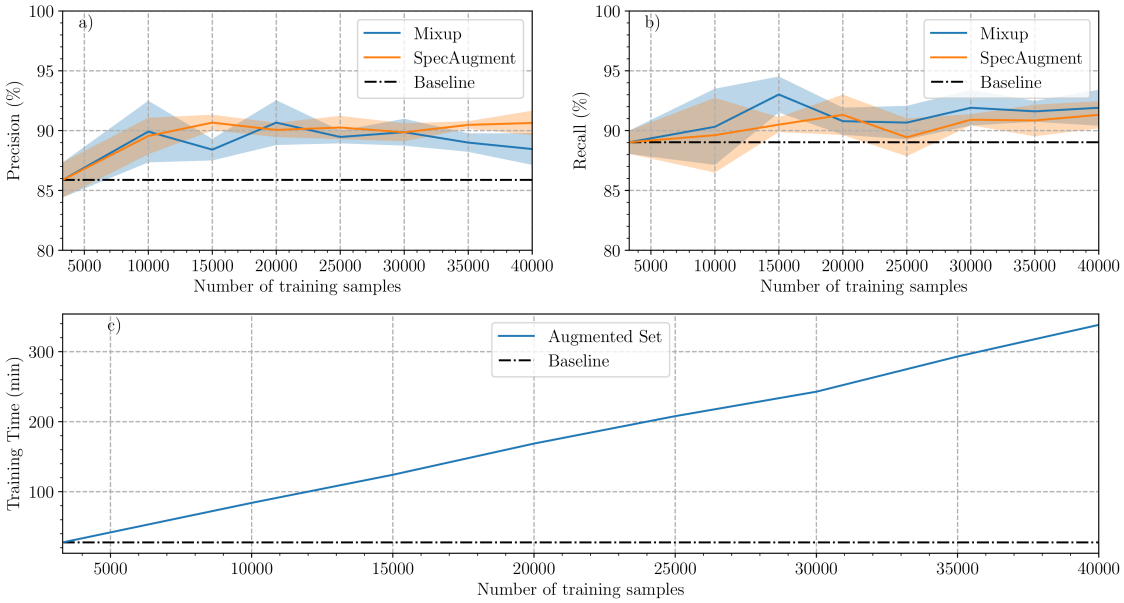

Figure 4.3: Classification performance (%) in terms of precision (a), recall (b) and training time (c) when training a ResNet on augmented datasets. The Figure displays the performance and training time when training on progressively larger augmented datasets using either SpecAugment or Mixup as the augmentation method. The Baseline is the full original dataset without augmentation.

4.4.2 Undersampled Dataset

We have also investigated the performance of the proposed methodology in a scenario of data scarcity by first undersampling our original dataset to establish baseline performance. With very low amounts of data available (200-400 samples), the model was not able to learn properly in order to generalize at all, and classifies almost every new sample as positive achieving nearly 100% recall and 50% precision. Furthermore, throughout the entire training procedure the model displays a substantially higher loss in the test set than in the training set when training with just 200 or 400 original samples. By increasing the amount of training data available, either original or augmented training data, the model is then able to learn to identify more effectively, and generalizes better to unseen data. For instance, when increasing the amount of original data available, we observe a gradual increase in performance, achieving 80.4% precision and 77.6% recall at 1000 original samples. It is worth noting that there are substantial differences in performance between the five models trained for each experiment when working with a small subsample of the original dataset. For instance,

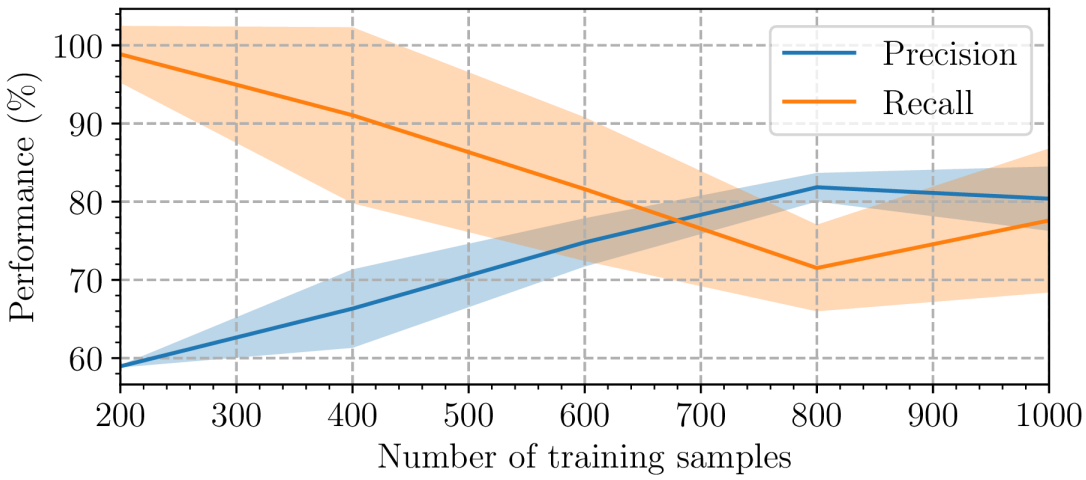

Figure 4.4: Baseline performance (%) in terms of precision and recall when training a ResNet on a subsample of the original dataset. The ResNet was trained with progressively more original data (starting from 200 original samples and adding 200 original samples at each step up to 1000 original samples).

with 1000 original samples we obtain a standard deviation of 4.1% for precision and 9.2% for recall as shown by the width of the shaded bands in the Figure 4.4. With 400 original samples available for training, we obtain a standard deviation of more than 10% on recall and 5% on precision for both Mixup and SpecAugment.

Next, we investigated the effects of data augmentation by progressively increasing the amount of augmented data available for training the DNN using either Mixup or SpecAugment as the data augmentation method (Figure 4.5). The dotted line represents the baseline performance when training with all the available original data.

We observe that the performance of the network is considerably improved by the data augmentation step, especially in the most extreme cases of data scarcity (200 and 400 samples). In these cases, instead of overfitting the training set and achieving a precision close to 50% and a recall close to 100%, the network achieves both a recall and precision close to 80% using just 200 samples, which corresponds to less than 10% of the available original training data. Using about a third of the original training dataset (1000 samples) and 6000 augmented samples, the process was consistently achieving recall levels close to the baseline, and precision even higher than the baseline for both Mixup and SpecAugment.

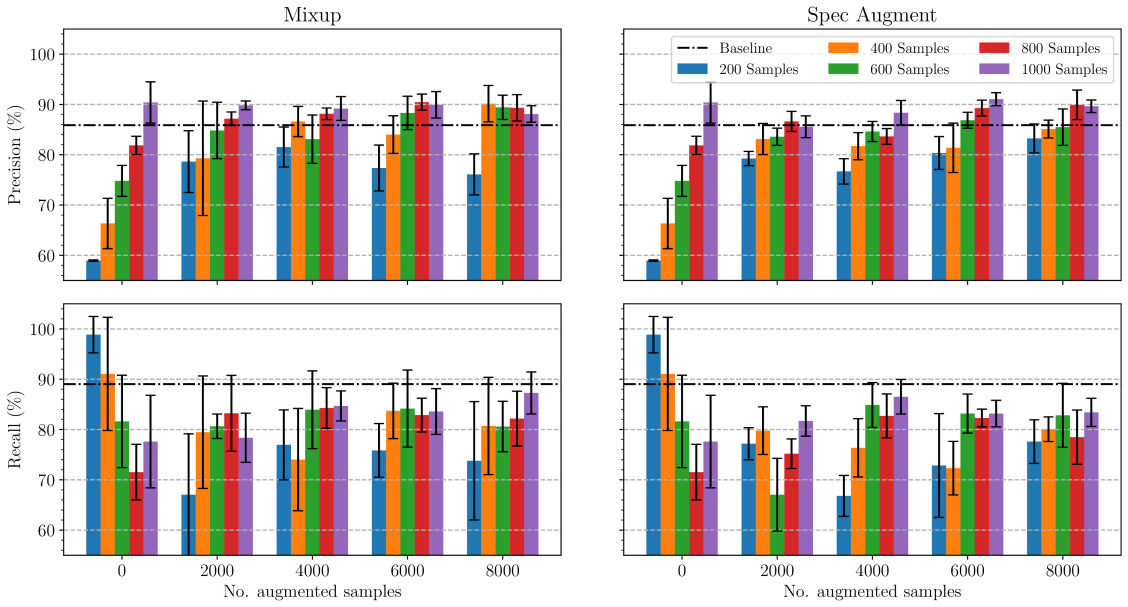

Figure 4.5: Classification performance (%) in terms of precision and recall when training a ResNet with progressively more augmented data (starting from 0 and adding 2000 samples at each step) generated with Mixup and SpecAugment using an undersampled dataset. Each bar represents the number of original samples and the error bars represents the standard deviation for that particular experiment.

4.5 Discussions

4.5.1 Full Dataset

With respect to the full dataset experiments, we found that both augmentation methods, Mixup and SpecAugment, contribute significantly to improving the performance of the network, and achieve comparable gains with the same number of augmented samples. This suggests that both methods are effective in increasing the diversity of a dataset under the proposed scenario, but do not provide any additional features to justify using one over the other. Moreover, substantial gains were already achieved with a moderate amount of augmented data. Mixup achieved a precision of 90.7% and a recall of 90.8%, while SpecAugment achieved a precision of 90.1% and a recall of 91.3%" compared to the baseline metrics of 89.0% and 85.9% for recall and precision respectively. Particularly in regards to the precision, adding a data augmentation step improved the model performance without compromising its recall. On the other hand, further inflating the size of the training set did not bring significant benefits to

justify the increased training time.

In general, based on the improved performance obtained by diversifying the training set, we argue that more complex data augmentation techniques such as generative methods based on DL models [88, 99] have the potential to significantly diversify the training set further improving the performance of a DNN. By generating new synthetic samples drawn from the dataset distribution these methods offer a way to unlock additional information from the dataset, which cannot easily be unlocked with traditional augmentation methods such as the ones used here. For instance, while the Mixup method can generate new audio samples by combining two waveforms, these new samples are generated in a very specific manner that does explore the full sample space as the Mixup algorithm does not modify the temporal position of a NARW upcall. Deep generative models however can learn to produce new samples with upcalls at any possible temporal position or frequency range naturally observed in the original dataset, expanding the effects of augmentation.

Finally, an important observation can be made that augmentation in general caused the DNN model to underfit the training data. This can be observed from the training curves displayed in Figure 4.6, where our original overfitting problem (indicated by the baseline curves) turned into an underfitting problem. In fact, we find that the loss obtained on the training set when augmenting with the Mixup method was even higher than the loss on the test set. This implies that this augmented training set is harder for the model to solve than the data contained in the test set.

There are several benefits to this. The first obvious one, is that the DNN was able to benefit from the increased data diversity by generalizing better to new unseen data. Second, in contrast to the usual problem of finding ways to deal with overfitting, underfitting can be more easily addressed by increasing the model complexity, for example by using deeper and wider networks, or by training for longer [108]. This has the potential to further increase the model‘s ability to generalize to unseen data. To verify this potential, we conducted an additional experiment by training our same DNN on the 20000 samples mixup augmented dataset for 100 additional epochs (200 epochs in total). We observe that the model trained for 200 epochs was able to lower the loss from 0.14 at 100 epochs to 0.12 at 200 epochs in the training set, with

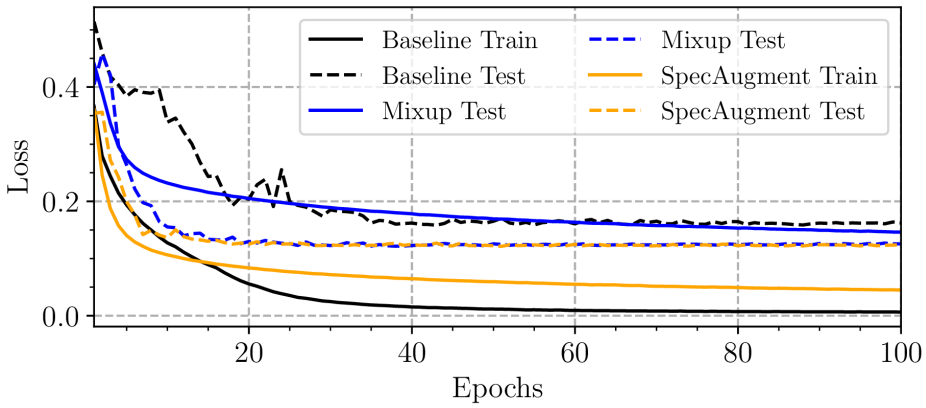

Figure 4.6: Training (straight lines) and test (dashed lines) loss curves of the model when trained on the base original training set (black lines), and on the augmented training set with Mixup (blue lines) and SpecAugment (orange lines) for 100 epochs.

no changes to the test set performance. This, in turn, implies that the model got better at classifying the harder dataset by training longer, however these gains did not translate into improved classification performance on the test set.

4.5.2 Undersampled Dataset

Concerning the undersampled dataset experiments, data augmentation greatly improved the performance of a model in a scenario where there was insufficient amounts of data to train a DNN. For instance, with 800 original samples (less than a third of the full original dataset) and 6000 augmented samples, the model achieved 89.3% precision and 82.3% recall with SpecAugment and 90.5% precision and 82.9% recall with Mixup whereas it only achieved 81.9% precision and 71.5% recall when trained only on the 800 original samples. Furthermore, we found that the DNN performed poorly under severe lack of data as 100% recall and 50% precision means that every test sample was being classified as positive. With data augmentation, even though the results remain considerably lower than the full baseline's, there has been a large improvement in providing a helpful classification.

Also, it was observed that access to more original data was crucial in lowering the standard deviation and it is evident that augmented data is not a perfect replacement for real data. For instance, the standard deviation obtained with 400 original samples

was almost always larger than the standard deviation obtained with 800 or 1000 samples, in all levels of augmentation. However, although the standard deviation for both precision and recall was often lower in the augmented set when compared to the unaugmented ones, access to more original data plays a larger factor in reducing the standard deviation on the performance metrics. Since both augmentation methods used in this study only transform or combine the available data, and we controlled other sources of variance such as initial training seed, the high standard deviation implies that they heavily depend on the initial draw of samples when building the undersampled set. That is, depending on the initial draw, the feature space present in the training set may not be sufficiently diverse for the model to properly learn, and the data augmentation methods studied will only be able to expand the existing feature space. On the other hand, the low standard deviation displayed on the full training set and its augmented versions shows that the feature space contained in the full training set is more diverse and the data augmentation techniques are able to capitalize on this diversity.

Chapter 5

Transfer Learning for PAM

The contents of this section were based on our work published in: Padovese, B., Frazao, F., Kirsebom, O. S., Evers, C., Beslin, W. A. M., Theriault, J., & Matwin, S. (2023). Adapting Deep Learning Models to New Acoustic Environments - A Case Study on the North Atlantic Right Whale Upcall. Ecological Informatics. p. 102169. [107]

In this chapter, we investigate adapting DL models trained to detect NARW upcalls in one acoustic environment to a new acoustic environment. As established previously, marine soundscapes are characterized by a vast diversity in environmental conditions, making it difficult to develop a universally applicable model that can effectively process data from different PAM systems, even if the task, such as detecting NARW vocalizations, remains the same. These systems may differ in their deployment locations, depths, and types, and consequently, the collected data may vary in terms of transient sounds or noise sources, ambient sounds, sound propagation characteristics, and hardware-induced system self-noise.

TL offers a promising approach to address these challenges by leveraging knowledge acquired in one or more source domains and applying it to a different target domain. Typically, the source model has been trained on vast amounts of data, allowing the network's layers to learn features that are useful for the given task. Ideally, some of these features would be generic enough to be shared across tasks, making TL an efficient way to transfer the knowledge learned in the source domain during the training process of a new model for the target domain. In this way, even small target datasets can benefit from the robust representations learned previously, and the target data only needs to adapt the new model to the new environment. In this context, it is noteworthy that only the source model and target dataset are necessary to initiate the TL process, which significantly enhances usability in cases where the source data is unavailable.

In this thesis, we have access to both large datasets with a few thousands of samples (the GOM dataset) and substantially smaller ones (the EMB dataset), which can represent the diverse range of data availability scenarios in underwater acoustic research. For instance, we can consider the GOM dataset as a source domain, where a DL model is trained on a large annotated dataset representing one acoustic environment. On the other hand, the EMB dataset can represents a target domain, which corresponds to a different acoustic environment to which we aim to adapt the model. By employing TL, we adapt the trained model to the target environment using substantially fewer annotated data than was initially required for training, offering insight into the practical application of TL for the proposed task of detecting marine mammals in different underwater environments with limit ted annotated data.

5.1 Materials and Methods

5.1.1 Datasets

This phase relies on all three datasets outlined in Section 3.2, namely the GOM, GSL, and EMB. Every subset of data from these datasets was employed. During this phase, the first four days of the GOM dataset were designated for training while the remaining three days were used for validation. A temporal split was also applied to the GSL data, reserving 85% for training and 15% for validation. These splits follow the protocol described in [44] for the GOM dataset and in Kirsebom *et al.* [65] and Section 4.1 for the GSL dataset. The validation data were used for confirming that model training had converged without overfitting. The first year of the EMB dataset (EMB_2015) was used for model adaptation while the second year (EMB_2016) was used for model testing. This split allowed us to adapt and test a model in a new acoustic environment while not having any temporal overlap between the data used for model adaptation and the data used for testing. For constructing the test set, 100 audio files were extracted from the EMB_2016 dataset and subjected to a full manual review by an expert to identify all NARW upcalls. The files have a combined duration of over 18 hours and span all 10 months of the monitoring effort. Of the 100 files that were extracted, 75 files were randomly selected from those that were flagged by the LFDCS algorithm as containing at least one true upcall, while 25 files

were selected at random from the remaining audio files that were not flagged by the LFDCS algorithm as containing an upcall. Table 5.1.1 provides an overview of the NARW upcalls distribution across each dataset and respective split.

We note that this process of testing the performance of the model on continuous data, as opposed to simply extracting segments and creating a test dataset as done in Chapter 4 provides a more realistic assessment of the performance of a model on real data. This is because the conventional method of testing on selected segments can be constrained by a limited variety of background environments, which may be chosen randomly, hand-picked, or flagged by another detection algorithm. Moreover, in real-world settings, marine mammal vocalizations, such as those of NARWs, are extremely sparse occurrences, and the composition of test sets often fails to represent the true ratio between background noise and vocalizations. In such cases, the performance demonstrated by the classifier in these test sets may not be representative of its performance on full-length recordings from the target environment. On the other hand, by evaluating a detector and classifier on full-length data, we can provide a much more realistic measure of detector performance in real-world scenarios.

Dataset	Total	Training	Validation	Adaptation	Testing
GOM	9,063	6,357	2,706	-	-
GSL	4331	3,715	616	-	-
EMB_2015	232	-	-	232	-
EMB_2016	504	-	-	-	504

Table 5.1: Total number of annotated NARW upcalls in each dataset as well as the number of upcalls reserved for the NARW detector's training, validation, adaptation and testing. A dash indicates that a particular dataset was not used for either training, validation, adaptation or testing. For a detailed description of the datasets, refer to Section 3.2.

5.1.2 Data Preparation

To generate the spectrograms and construct the training and adaptation datasets, we employed the methodology for extracting positive and negative samples and the spectrogram computation steps outlined in Section 3.3. For the EMB_2015 dataset only, due to the limited amount of upcall annotations available for TL, multiple 3-second segments were extracted from each annotation by shifting the spectrogram

window by steps of 0.5 seconds forward and backward in time with respect to the midpoint of the upcall. Each upcall sample is represented $N \geq 1$ times in the training set, but the value of N is not the same for all samples, as only spectrogram views that included at least 50% of the upcall were extracted. In this way, instead of having a single view that is usually centralized in the midpoint of the upcall in time, we produced additional views where the vocalization is slightly shifted in either direction. In addition to increasing the size of the dataset, the translated spectrogram views encourage the network to learn a more generalized and translation-invariant representation of the upcall. The resulting number of positive and negative samples extracted from each dataset are shown in Table 5.1.2. For the EMB_2016 testing dataset, no segments were extracted as our evaluation approach involved testing on the complete continuous data.

Dataset	**Training**			**Adaptation**		
	Total	Postive	Negative	Total	Positive	Negative
GOM	12,714	6,357	6,357	-	-	-
GSL	6,952	3,715	3,237	-	-	-
EMB_2015	-	-	-	7,403	1,392	6,011

Table 5.2: Number of positive and negative samples used for training and adapting the NARW upcall detector. For details on how the positive and negative samples were extracted see Section 3.3. Note that the number of positive samples extracted from the EMB_2015 dataset was inflated by extracting multiple, time-shifted views of the same upcall.

5.1.3 Neural Network Architecture

In addition to the ResNet-18 architecture described in Section 3.4, we employed another commonly used CNN architecture to classify NARW ucpcalls, namely the VGG-19 [131] which was implemented with batch normalization [57]. This specific architecture begins with an initial convolutional layer with 64 filters, followed by 4 sets of convolutional blocks. Each set contains a varying number of convolutional layers with 3x3 kernels, and the depth of the network increases by doubling the number of filters after each block. In this implementation, the first and second blocks have 2 convolutional layers each, the third block has 3 convolutional layers, and the fourth and final block has 8 convolutional layers, for a total of 16 layers. Each convolutional

layer is accompanied by batch normalization and employs ReLUs [96] as the activation function. Between the blocks, max-pooling layers are employed. After the final set of convolutional blocks, the network utilizes a final batch normalization [57] layer and global average pooling [78]. The convolutional layers are then followed by 2 dense layers. At the end, as with the ResNet-18, a fully connected layer with a softmax function produces a score for each of the two possible classifications (upcall present or absent). These scores, ranging from 0 to 1, represent the network's confidence in each classification, and their sum equals 1.

5.1.4 NARW Detection Tool

As DL algorithms are increasingly being used in marine bioacoustics, there is a growing need for ready-to-use tools that allow researchers to apply and adapt DL models to their own data as well as integrate them with various software. In light of this, the Python tool we created provides a high-level interface for developing DL-based detectors through a command-line interface (CLI). By providing concise options for performing common tasks required during the detector development stage, our tool allows researchers to design and adopt DL methods for the detection and classification of NARW vocalizations. Additionally, the tool adopts a standard output format that makes it simpler for researchers to transition their work to a finalized model that is open to share, use, and adapt.

The tool is organized into sub-modules, serving users with varying needs, who have a high-level understanding of ML and DL concepts without requiring any programming skills. These sub-modules can be incorporated into the detector development workflow on an as-needed basis. Users with more advanced programming knowledge can also integrate the modules into external applications.

5.1.5 LFDCS Performance

To set a performance benchmark for comparison with the DNNs trained in this phase, we initially evaluated the LFDCS detector, which was first used in [32] to generate preliminary annotations. It is worth recalling that the annotations from [32] only encompass the validated detections produced by the LFDCS algorithm. To our knowledge, a full manual analysis has not been previously carried out on the Emerald

Basin recordings. For performance metric calculations, we assembled a test dataset consisting of 75 files containing at least one verified true upcall, as detailed in Section 5.1.1, and an additional 25 files that were initially flagged by the LFDCS detector as containing a NARW upcall but were later verified as false positives by an expert. Performance evaluation metrics, including precision, recall, and FPR per hour, were calculated with respect to the positive class, as defined in Section 3.5. This step enables a direct comparison between our methodology and a traditional detection algorithm still widely used for the automated classification of NARW vocalizations [60, 32].

We found that the LFDCS model achieved a precision of 0.52 and a recall of 0.41. Across all 100 test files, it generated 140 false positives, resulting in an FPR per hour of 7.56 over 18.5 hours of recordings. It should be noted that the composition of our test dataset is skewed towards true positives, as 75% of the files were initially flagged by the LFDCS detector for containing genuine upcalls, while the remaining 25% were files identified as false positives. This is, however, acceptable in the present study as the aim of this step is to provide a performance benchmark against which to compare our deep learning model. A comparison with the methods implemented in this phase can be seen in Section 5.2.

5.1.6 Experimental Setup

We conducted three experiments in order to evaluate the effectiveness of model adaptation through TL. For all three experiments, the models were evaluated on the EMB_2016 data. This dataset was reserved for testing purposes only and was not used for training or TL. Similarly to phase one, all experiments were conducted on a workstation with a NVIDIA TITAN V GPU with 12 GB of memory using the NARW detection tool developed as part of this work.

For the first experiment, we trained the two DNN architectures described in Sections 3.4 and 5.1.3 on the GOM and GSL datasets with a batch size of 64 samples and learning rate of 0.001 for 40 epochs using the ADAM [64] optimization algorithm. We determined that 40 epochs were enough for the optimization to converge without overfitting. In the second experiment, we used the EMB_2015 dataset to adapt

the models from the first experiment to the EMB acoustic environment. When fine-tuning the models to the new data, we decreased the learning rate by a factor of 10 to 0.0001 and reduced the number of training epochs to 20. The reduced learning rate and number of training epochs reflects the fine-tuning process of TL. Finally, in the third experiment, to demonstrate that the EMB_2015 data are insufficient to produce a useful detector without model adaptation, we established a baseline performance by training the two DNN architectures on the EMB_2015 dataset only.

Samples were classified as 'positive' if the positive-class score computed by the model exceeded a certain threshold value, and were classified as 'negative' otherwise. Model evaluation was conducted by computing the precision, recall and FPR with respect to the positive class. For more details regarding the metrics, refeer to Section 3.5.

In addition to the performance metrics described above, we selected the ResNet model to display the embeddings produced by the model before and after model adaptation. The embeddings are a low-dimensional representation of the data, and were generated by extracting the output of the last feature extraction layer of the network. The output is a collection of features produced by the network that are then reduced to two dimensions with the t-SNE algorithm [138]. When plotted in a 2-d space, similar data points appear near one another, ideally forming distinct clusters that correspond to each class. This approach allows us to better visualize if the features learned by the DNN have strong discriminatory power between the two classes.

5.2 Results

In Table 5.2, we present a comparative summary of each model's performance in terms of precision recall and FPR per hour at a fixed detection threshold of 0.5 for easier comparison with the LFDCS detector. Overall, both DL models, before and after TL, outperformed the LFDCS detector in terms of both recall and precision with the ResNet model after TL achieving the highest recall and precision at 0.87 and 0.85, respectively. Conversely, the baseline models, trained on limited data, exhibited high FPRs per hour of up to 85 for the baseline VGG model and 53 for the baseline ResNet model. In contrast, the LFDCS detector, while achieving moderate levels of recall

and precision, maintained a more balanced performance with fewer than ~ 8 FP per hour.

Model	Precision	Recall	FPR (h)
LFDCS	0.52	0.41	7.56
Baseline - VGG	0.2	0.79	85
Baseline - ResNet	0.28	0.78	53
VGG - Before TL	0.71	0.57	6.32
ResNet - Before TL	0.80	0.69	4.4
VGG - After TL	0.58	0.84	16.3
ResNet - After TL	**0.85**	**0.87**	**4**

Table 5.3: Precision, recall and FPR per hour of each model's performance at a fixed detection threshold of 0.5.

Figures 5.1 and 5.2 summarize the performance of the various models on the EMB_2016 test set. In Figure 5.1 we show precision against recall, while in Figure 5.2 we show FPR against recall. The colors blue and orange are used to distinguish the two different architectures used (i.e., ResNet-18 and VGG-19) and the green color indicates the LFDCS algorithm. Dotted curves show the baseline performance of the model trained exclusively on the EMB_2015 dataset. Finally, dashed and solid curves show the performance of the models trained on the GOM and GSL datasets before and after adaptation, respectively.

At a fixed recall of 0.41, the LFDCS detector achieved a similar performance to that of the baseline models, reaching a precision of 0.52. This corresponded to an FPR per hour of ~ 7.5, which is lower than the VGG model after TL, albeit at the cost of lower recall. For context, the VGG model was capable of achieving a recall of up to 0.9 while maintaining an FPR of approximately 10. We note that the ResNet models both before and after TL achieved a better performance than the LFDCS detector both in terms of precision and recall at all detection thresholds.

After adaptation, both the VGG and ResNet models outperformed their source model on the EMB_2016 recordings at high recall levels. Both models showed a large improvement in recall, with the recall of the ResNet model improving from 0.65 to ~ 0.90 for a precision of 0.85, and the recall of the VGG model improving from 0.74 to ~ 0.86 for a precision of 0.60. Overall, the ResNet model after adaption had the highest recall (~ 0.95) with the lowest FPR of below 9 per hour out of all the models.

The performance of the baseline models trained only on the EMB_2015 data were substantially inferior to the models trained on the GOM and GSL datasets, both before and after adaptation. While the baseline model had a recall comparable to that of the models trained on the GOM and GSL datasets without adaptation, it produced a large number of false positives, leading to a low precision for any given recall and a high FPR.

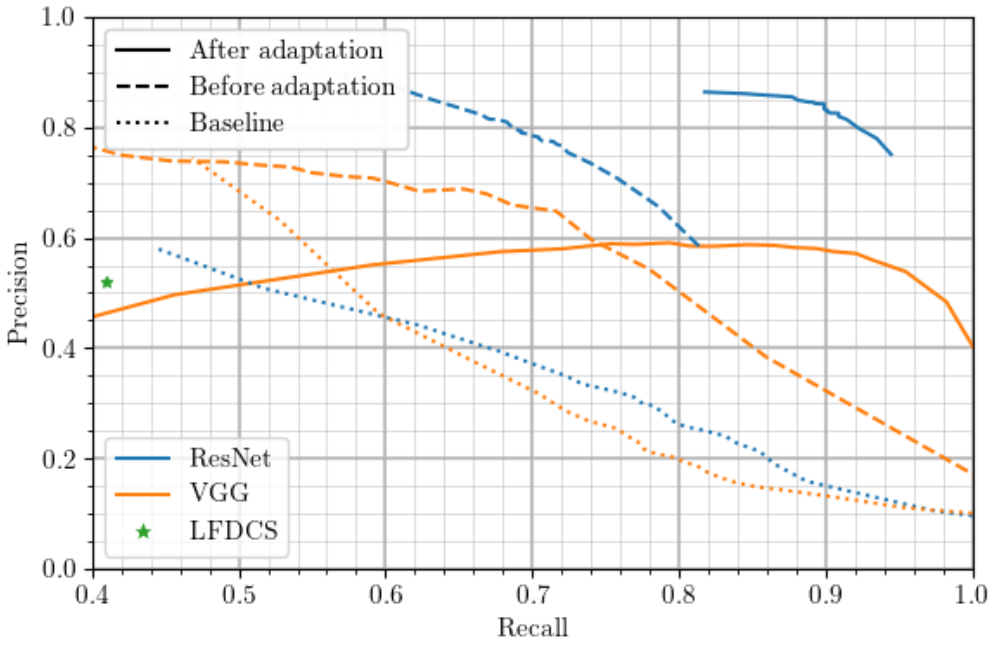

Figure 5.1: Precision vs. recall on the EMB_2016 test set for the various models investigated in this study. Results obtained with the ResNet-18 architecture are shown in blue while results obtained with VGG-19 are shown in orange. Dotted curves show the baseline performance of the models trained only on the EMB_2015 dataset while dashed and solid curves show the performance of the models trained on the GOM and GSL datasets before and after adaptation, respectively. The green star denotes the LFDCS detector performance.

In general, model adapation improved the recall without significantly worsening neither the precision nor the FPR. The VGG model showed a constant precision of $\sim$ 0.6 for recalls below $\sim$ 0.9, with the FPR increasing considerably only when the recall approached 1. The ResNet model displayed a precision above 0.75 (corresponding to a FPR of less than 9 per hour) for any given recall. At a recall of $\sim$ 0.81, the ResNet model achieved the lowest FPR of only 3, corresponding to a precision of $\sim$ 0.87.

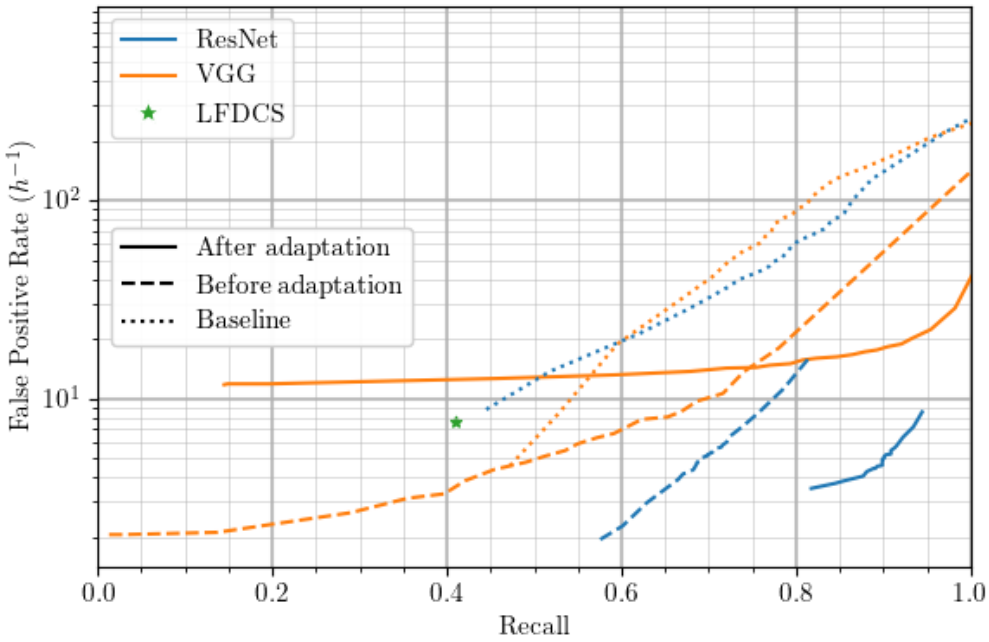

Figure 5.2: Precision vs FPR on the EMB_2016 test set for the various models investigated in this study. Results obtained with the ResNet-18 architecture are shown in blue while results obtained with VGG-19 are shown in orange. Dotted curves show the baseline performance of the models trained only on the EMB_2015 dataset while dashed and solid curves show the performance of the models trained on the GOM and GSL datasets before and after adaptation, respectively. The green star denotes the LFDCS detector performance.

Figure 5.3 further reinforces our findings where we display the two-dimensional embeddings of the ResNet model predictions on the EMB_2016 data before (top), and after (bottom) the adaptation procedure. While the positive and negative classes already form two distinct clusters before the adaptation step, the clusters become more distinct after adapting the model to the EMB acoustic environment through TL. We note that this picture is consistent with the reasonable classification performance obtained already before the adaptation step and the improvement observed upon applying TL.

We used the Python tool that we developed to carry out the three experiments described in this study. The tool is organized into 5 sub-modules, which allow the user to work through 5 major steps required for developing a DL detector for underwater bioacoustics. These steps are: 1) creating training and test datasets from raw

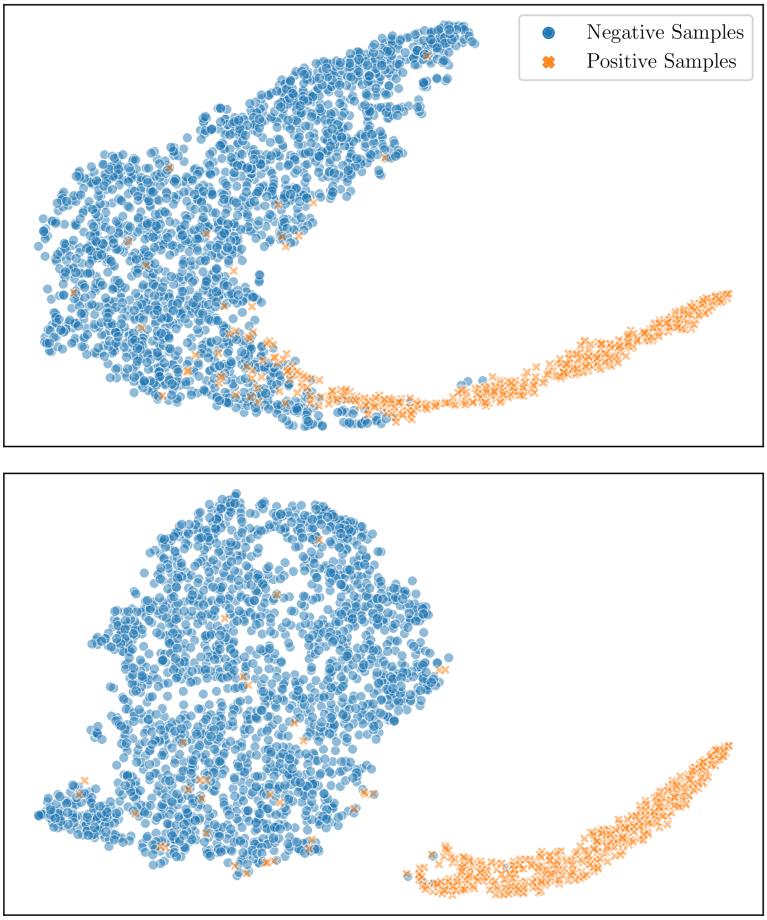

Figure 5.3: Two-dimensional t-SNE embeddings of the predictions of the ResNet on the EMB_2016 data before (top) and after (bottom) model adaptation. The 'x' symbol represents positive samples while the 'o' symbol represents negative samples. To produce the embeddings we extracted the features produced by the last feature extraction layer of the network and reduced their dimension with the t-SNE algorithm.

acoustic data using different audio representations (i.e., spectrograms, cepstrogram, etc.); 2) training a DL model to detect and classify underwater acoustic signals; 3) evaluating the model on fully-annotated test datasets and reporting standard performance metrics such as precision, recall and FPR; 4) running the model on a set of audio files; and 5) adapting the model to a new underwater acoustic environment. Each sub-module also includes several options, such as the ability to define a detection threshold in sub-module 4. The tool is provided as an open-source package in a GitLab repository[1] and includes relevant documentation, instructions and examples.

[1]GitLab repository: https://git-dev.cs.dal.ca/meridian/NARW_detection_tool

5.3 Discussion

The quality of DL models is dependent on the availability of large quantities of annotated data. In this work, given limited number of annotated upcalls in the EMB_2015 dataset, the ability to leverage the knowledge learned from the considerably larger GSL and GOM datasets through model adaptation allowed a DL model to be used in a scenario where it would have otherwise performed poorly. This is evidenced by the performance of the baseline models trained only on the EMB_2015 data, which were not able to generalize well to unseen samples from the same acoustic environment. In contrast, the performance of the source models trained on the larger GOM and GSL datasets was already an improvement over the baseline models before model adaptation, even without access to data from the EMB testing environment. This indicates that the features learned from the source environments already formed a reasonably robust representation of the positive and negative classes in the testing environment. By adapting the source models to the testing environment, we were able to verify that the adapted models had a substantial improvement in performance. At a FPR below $\sim$ 5 per hour, the ResNet model retrieved $\sim$ 85% of the upcalls, improving from $\sim$ 70%, while at $\sim$ 11 FPR, the VGG network retrieved $\sim$ 87% of the upcalls, up from $\sim$ 70%.

The distinct division between the positive and negative classes in the adapted ResNet model 5.3, suggests that the features learned by the DNN have strong discriminatory power between the two classes. In other words, the DNN has learned distinct representations for each class even before the classification layers. This finding is significant as it enables the use of these representations as input for clustering methods and other post-processing techniques. In this context, we highlight the works by [103] that cluster the features learned by an autoencoder (a DL method) to distinguish between fish and whale calls in coral reefs, and [9] that adopt a fully DL-based unsupervised feature learning approach to cluster and distinguish between killer whale (*Orcinus orca*) call types.

When comparing the DL models to a traditional detection algorithm like the LFDCS detector, it is evident that the DL models, both before and after TL, outperformed the LFDCS detector across various metrics. Notably, even the models before TL surpassed the performance of the LFDCS detector. This suggests that the

features learned by the source models—trained on extensive datasets—were highly effective at generalizing to the target task. Conversely, the limitations of DL-based approaches become apparent when we contrast the LFDCS detector's performance with that of the baseline models, which were trained under data-limited conditions. Although these baseline models achieved higher overall recalls, these gains were offset by a significant decrease in precision and a marked rise in the FPR per hour. This indicates that the models were biased toward classifying the majority of samples as positive. In contrast, the LFDCS detector maintained a much higher level of precision while capturing slightly less than half of the upcalls, providing a more balanced performance which can be beneficial in a real-world context.

Higher rates of false positives can substantially increase the amount of time analysts must spend manually validating detector outputs. Therefore, a lower FPR is often desirable, even if it results in a lower recall. A higher recall, however, may be prioritized in situations where near real-time detection is required, such as when monitoring a species for conservation efforts. While assigning an exact value to a desirable precision and recall can be challenging due to factors such as the analysis objective, and the target location and species, we can illustrate the cost of false positives in an analysis with the following example: Consider a one-month monitoring effort with one recorder, totaling 744 hours. Employing the ResNet model presented in this work, which generates an average of 5 false positives per hour, would result in approximately 3,740 false positives requiring validation. As the monitoring duration and number of recorders increases, the number of false positives to be validated will also rise. In this context, Shiu *et al.* [127] defines a useful detector as one that produces fewer than an average of 20 false positives per hour of recording. Similarly, both Simard *et al.* [130] and Kirsebom *et al.* [65] tuned their detectors to prioritize higher precision over recall, with recall levels reaching as low as 62%.

In our work, the best model, ResNet, displayed substantially higher recall after model adaptation without losing any precision, which indicates that the model was able to generalize well to the new acoustic environment. In addition, we examined the false positives from this model and found that the majority of them were signals that were similarly shaped like an upcall. This suggests that the model is able to accurately

detect the characteristics of an upcall against the background environment, but struggles against similar shaped signals originating from other sources. Fig. 5.4 displays spectrograms of the model's classifications, illustrating both correct (true positives) and incorrect (false positives) detections. In this context, humpback whales, in particular, produce vocalizations that are acoustically similar to NARW vocalizations and have been reported to trigger large numbers of false positives [26, 32]. Additionally, humpback whales are often more vocally active than right whales [35, 32]. This elevated vocal activity could result in a higher number of false positives when the objective is to detect NARW calls, thereby complicating the detection efforts further.

Another challenge in developing detectors capable of differentiating between the upcall vocalizations of right whales and humpback whales is the scarcity of annotated data that covers both species from a single geographical location. Typically, annotation efforts within a specific region or data collection effort focus on a singular species or type of vocalization, resulting in datasets that include annotations for one species but omit others. Consequently, when DL-based models are trained to distinguish between the two species using these incomplete datasets, they often end up learning to discriminate based on the background noise characteristics unique to each dataset's environment, rather than learning accurate representations of each type of vocalization. While several works have proposed multi-species classifiers that include both humpback and right whale classes [136], only a few studies [32] focus on the specific challenge of developing a robust architecture capable of effectively distinguishing between the vocalizations of these two species. For instance Durette-Motin *et al.* [32] employ prior knowledge that NARW upcalls tend to occur irregularly, whereas humpback sounds are often repetitive and may be associated with songs, particularly from fall to spring [142], to help distinguish both species vocalizations. They use this knowledge to automatically discard automated NARW detections that occur within a 24-hour period of detected humpback whale presence, provided that no other characteristics identifying the sound as NARW are found. Such solutions however, often demands highly specialized expertise and can be difficult to implement. These limitations highlights the importance of expanding datasets to include vocalizations from other species in order to build more robust detectors.

Another important limitation of the current work is that all recordings used in this

study were collected North of the Gulf of Maine in NARW feeding or migration areas. Future research should assess how the methods presented in this study perform on data collected from different environments and time periods, such as NARW calving grounds off the southeast coast of the U.S.

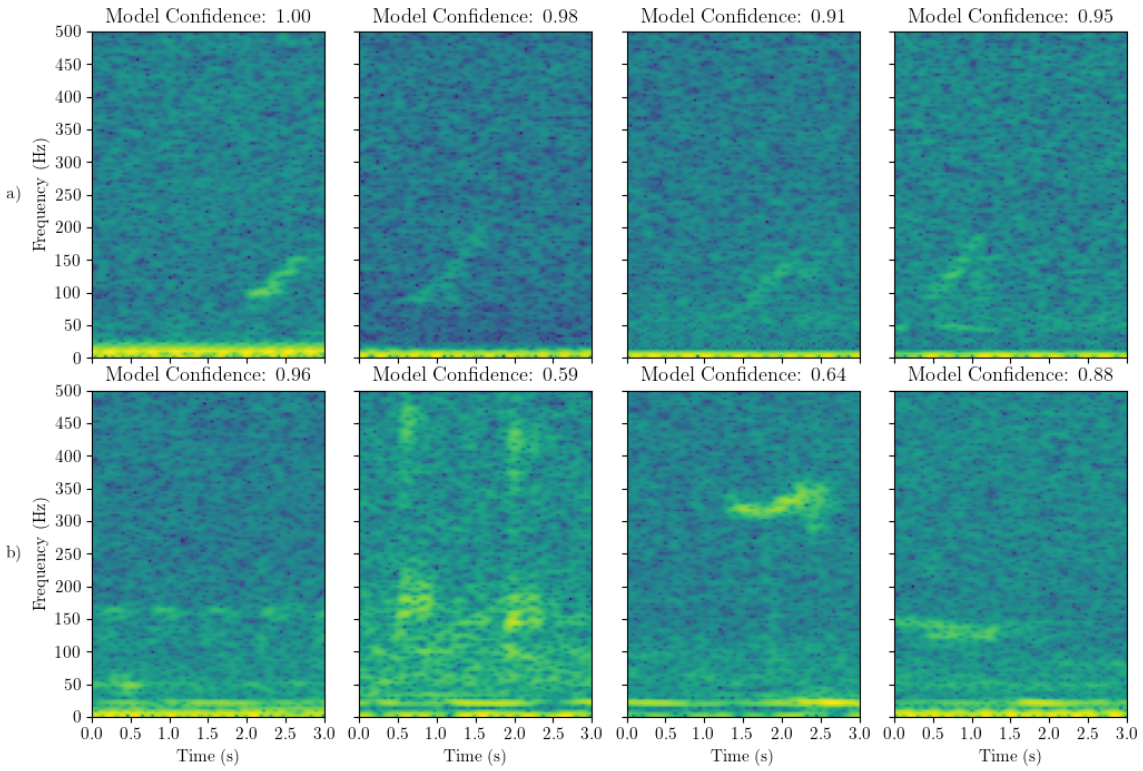

Figure 5.4: Spectrograms samples illustrating the model's classification performance on NARW upcalls. a) The first row displays four true positive examples where the model successfully detects and classifies upcalls. b) The second row presents four false positive examples, showcasing instances where the model misclassifies other acoustic signals or noise as upcalls. Panel titles indicate the models confidence in the detection.

The use of a CLI tool for creating DL-based acoustic detectors and classifiers facilitates the adoption of the methodology described in this study as well as allows users to compare and replicate the results. In its current form, the tool is agnostic to vocalization types and species, to the audio representation used (e.g., spectrogram or cepstogram) and to the type of CNN architecture chosen (e.g., VGG or ResNet), but still makes certain assumptions about the model, most importantly that it is a binary detector (i.e., only two classes). Another assumption made by the tool include

it being compatible with CNN architectures only, but not accepting sequence models such as RNNs. These assumptions limit the range of possible applications and DL-approaches to marine mammal acoustic detection but are limitations we intend to address in the future.

Chapter 6

Synthetic NARW Upcalls for Model Training

As established in previous chapters, obtaining large quantities of annotated vocalizations for training a DNN to detect and classify marine mammals is a costly and time consuming endeavor. In most situations, there is insufficient training data even after an initial annotation effort due to the sparse occurrence of signals of interest in the data. While the data augmentation techniques presented in Chapter 4 can greatly increase the performance of models in a context of data scarcity, they are also limited by the availability of sufficient annotated samples to apply the transformations to.

In this thesis, we explore mimicking NARW calls by generating synthetic signals that are acoustically similar to real upcalls. The motivation behind this approach is to train a DNN that is capable of achieving reasonable detection performance without feeding it any real vocalization. By synthesizing calls artificially one can augment a dataset that has yet to be analysed and reduce the need of having annotated data for training DNNs. In this way, researches have the option of validating the model's detections without the need for manually reviewing the recordings in search of vocalizations. To generate realistic upcalls, acoustic propagation modelling was used to transmit the synthetic signals through a virtual environment and simulate how the signals would behave in the recording environment as they propagate through the water. This approach enables modelling different underwater environments and multiple scenarios for the same environment, greatly expanding the size and diversity of a training dataset [12]. Finally, to make the synthetic upcalls even more realistic, we embed them in real background samples extracted from the environment. During this process, we scale the amplitude of the synthetic upcall relative to the ambient noise background to produce embedded calls with varying SNRs. This is important because a SNR mismatch between a sample containing an synthetic upcall embedded in the real ambient noise background, compared to a real upcall in the ambient noise background, can significantly impact the detection performance of a DNN trained on

synthetic data when applied to real data. Therefore, by achieving similar SNR levels as those found in real upcalls samples, we ensure that the synthetic upcall samples more closely resemble natural upcalls allowing the DNN to learn to detect them more accurately.

By combining synthetic calls, acoustic propagation modeling, and environment embedding, a wide range of upcall shapes and characteristics can be generated. This process results in a diverse dataset, providing an increased variability for training a DNN to detect and classify real upcalls, and subsequently saving valuable time that would otherwise be spent on analyzing the data. Another advantage of this approach is that researchers have complete control over the modelling parameters and can quickly fine tune any variable to reflect a specific need. A model trained on this data is therefore already tailored to the specific vocalization and environment of interest. Additionally, while the methodology was proposed to produce a training dataset without any annotations, it can be equally applied as a data augmentation step to an already existing dataset.

In this chapter, we experiment with training a DNN to detect NARW in the Gulf of Saint Lawrence using a training dataset produced with synthetic NARW upcalls propagated through a virtual environment representing the Gulf. We verify whether this DNN can be used effectively as a detector, extracting a high number of true detections while producing a low number of false detections. To our knowledge, this is the first work that combines both generation of synthetic calls with sound propagation modelling to produce a training dataset for DL-based detection and classification of marine mammals.

6.1 Materials and Methods

6.1.1 Datasets

To evaluate the effects of using synthetic data in a controlled setting, this chapter relies only on the GSL dataset described in Section 3.2. The GSL dataset was chosen for this stage of the research due to its well-documented data collection protocol [65] and the comprehensive description of the Gulf of St. Lawrence's acoustic properties in several works [80, 4]. These factors enable a precise definition of the parameters for modeling

the acoustic propagation of synthetic signals within the virtual environment. In order to isolate the recording environment to a specific setting and recording equipment, only a subset of this dataset, the data collected through the bottom deployments, was used. We recall that the bottom deployments were moored approximately 5 m above the seafloor and were split into multiple 1 kHz 3-seconds long clips (GSL_B), and 50, 30-minute long recordings sampled at 32 kHz (GSL_B*) amounting to 25 hours of continuous data. Both subsets were fully annotated for NARW upcalls. The GSL_B dataset contained 3,888 clips, of which 1,514 were negative samples and 2,374 were positive samples. Unlike in Chapter 4, this dataset was not divided into training and validation sets. For further details on the annotations and data collection process, refer to Section 3.2.

In this study, the 30-minute recordings were utilized for testing the DNN on continuous data and as a source of negatives for synthetic upcall generation and training. Specifically, this dataset served to simulate a scenario where no annotations are available as is typical in real-world situations. In this scenario, it is still possible to leverage randomly extracted segments of background noise for both synthetic generation and composing the negatives of a training sample. In this simulation, the DNN trained on synthetic data is then employed to detect any upcalls within the dataset. Meanwhile, the clips dataset was used to establish the baseline performance and for adapting a DNN that was initially trained on synthetic data, allowing us to observe the improvement in performance as we incrementally introduce real data.

6.1.2 Data Preparation

In this chapter, we adhere to the same spectrogram computation process outlined in Section 3.3. To create the synthetic dataset, for the positive set, we generate N synthetic samples, where the value of N is determined in the experimental protocol outlined in Section 6.1.7. The negative samples are acquired by randomly extracting 3-second segments from the 30-minute recordings dataset, with the number of background samples extracted equal to N. This process follows the same methodology described in Section 3.3.

To create a balanced dataset of the GSL_B clips dataset, we aimed to include an equal number of positive and negative samples. As there were initially 1,514 negative

samples, we supplemented this count by randomly extracting 860 additional 3-second segments from the 30-minute recordings dataset while avoiding the upcall annotations to match the 2,374 positive samples. This resulted in a total of 2,374 negative samples and an overall dataset comprising 4,748 samples. Table 6.1.2 displays the partitioning of real and synthetic data for both the synthetic dataset and the GSL_B clips dataset.

Dataset	**Synthetic Positives**	**Real Positives**	**Real Negatives**
Synthetic	N	-	N
GSL_B Clips	-	2,374	2,374

Table 6.1: Number of training samples in the two datasets, GSL_B clips and synthethic dataset, utilized in this thesis, showcasing the distribution of synthetic positives, real positives, and real negatives within each dataset. For details on how the synthetic samples were generated, see Sections 6.1.3, 6.1.4, 6.1.5 and 6.1.6.

6.1.3 Artificial Generation of Synthetic Upcalls

As previously stated, NARW upcalls were synthetically generated to compose a training dataset for training a DNN. The first step in generating synthetic calls of marine mammals is understanding their acoustic properties. We recall that the NARW upcall is a stereotyped low-frequency upsweep (typical frequency range 100 – 300 Hz) that lasts approximately one second (Section 3.1). Figure 6.1 presents various spectrogram representations illustrating the diverse shapes that NARW upcalls can exhibit. While this definition of a typical NARW upcall is often cited in the literature [65, 127, 40], it is natural to find considerable variance in the duration, frequency range, and pattern of upcalls. Furthermore, external factors such as transmission loss and interference from anthropogenic, biogenic, and environmental sources can further distort the signal, resulting in a wide variety of similar yet distinct upcalls both acoustically and visually through spectrograms. For instance, all of the upcalls displayed in Figure 6.1 were recorded at some distance from the vocalizing whale. As a result, the calls may have been significantly modified by propagation effects en-route from the whale to the hydrophone. Therefore, the variation we observe in these upcalls is a combination of intrinsic variation as well as propagation variation.

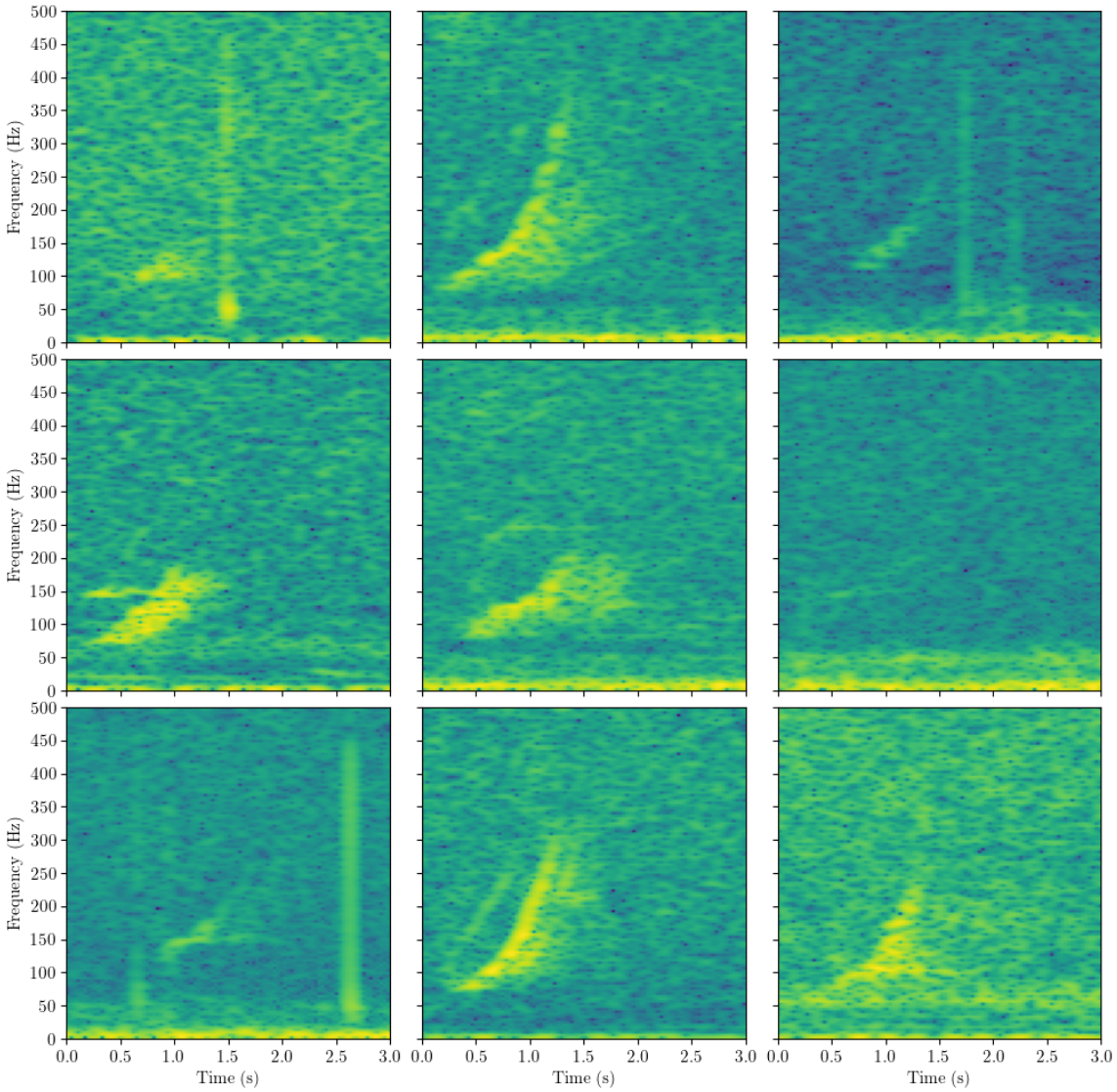

Figure 6.1: A collection of nine spectrograms displaying the diverse shapes and patterns of NARW upcalls, showcasing the natural variability in their acoustic properties.

In an ideal scenario, synthetically generated upcalls would be indistinguishable from natural upcalls, however, due to the complex nature of NARW upcalls and the various external factors that can impact their acoustic properties, it is impossible for an algorithm to anticipate all possible variations when generating synthetic upcalls. To establish a known starting point, we generated "clean" synthetic signals that have not been subjected to any kind interference or propagation effect. This provided us with a basis for experimentation with propagation effects, which will be discussed in the next section. By starting from a "clean" signal, we were able to more accurately

control the effects of propagation and better evaluate their impact on the performance of the DNN.

The algorithm to generate the "clean" synthetic signals was based on two factors, the literature definition described in this section and visual analysis of spectrograms of high SNR upcall samples drawn from the datasbase. In this thesis, the synthetic upcall is a sine wave that changes its frequency between f_{start} and f_{end} in T seconds to achieve an upsweep pattern. Motivated by the goal that these synthetic upcalls should have similar variations in the duration, frequency range, and pattern of natural upcalls, we developed the following policy to synthesize the upcall:

1. To generate a waveform that is D seconds long, a duration T between $[T_{start}, T_{end}]$ is first randomly selected for the synthetic signal. Then, a starting time P for the upcall is randomly chosen within the interval $[0, D - T/2]$. Note, that the upper interval limit of P allows for the creation of a synthetic upcall that is only half present in the generated waveform. This helps the DNN to learn a representation of the upcall in cases where it is only partially present in the spectrogram view, improving the robustness of the model.

2. The frequency upsweep pattern is generated by linear interpolation of four time-frequency points $(t_1, f_1), ..., (t_4, f_4)$, where $t_1 = P$, $t_2 = P + T\tau_1$, $t_3 = P + T\tau_2$, and $t_4 = P + T$. The values of $f_1, .., f_4$, τ_1 and τ_2 are randomly selected from user-defined intervals. The linear interpolation is used to create a function to determine the frequency f of the signal at any time point t using the equation,

$$f = f(t) \tag{6.1}$$

3. The phase of the signal is then given by,

$$\phi = 2\pi \int f(t)dt \tag{6.2}$$

where dt is the time step between two consecutive samples. Finally, the synthetic signal S can be generated with,

$$S = A \sin \phi \tag{6.3}$$

where A is the amplitude.

To ensure consistency between the synthetic and real data used in this thesis, all synthetic samples were generated with a sampling rate of 1000 Hz. The parameters for the policy used in this thesis are summarized in Table 6.1.3.

Parameter	**Value**
D	3
T_{start}	0.8
T_{end}	1.2
f_1	[80..105]
f_2	[115..135]
f_3	[160..195]
f_4	[230..290]
τ_1	[0.25,0.45]
τ_2	[0.6,0.8]

Table 6.2: User defined parameters used in this thesis to generate the synthetic upcalls.

While the goal of generating synthetic signals was to produce signals that closely resembled natural upcalls, the parameter values were not extensively explored to find an optimal combination. For instance, one possible approach to further enhance the realism of synthetic signals could involve experimenting with time-dependent modulation of the amplitude, where A varies slowly over time. However, this idea was considered beyond the scope of this work and could be explored in future research. Instead, the focus was on using the synthetic upcall samples to train a DNN to detect real upcalls. Therefore, some synthetic signals may appear slightly unnatural, but they are still suitable for DNN training. In fact, these quasi-random deviations are desirable for DNN training because they help the network learn a more general representation of the upcall [65]. In Figure 6.2, we present four distinct examples of synthetic upcall variations generated using the proposed algorithm.

6.1.4 Acoustic Propagating Modelling

As established in the literature review, underwater environmental conditions can significantly impact how sound propagates through the water and, consequently,

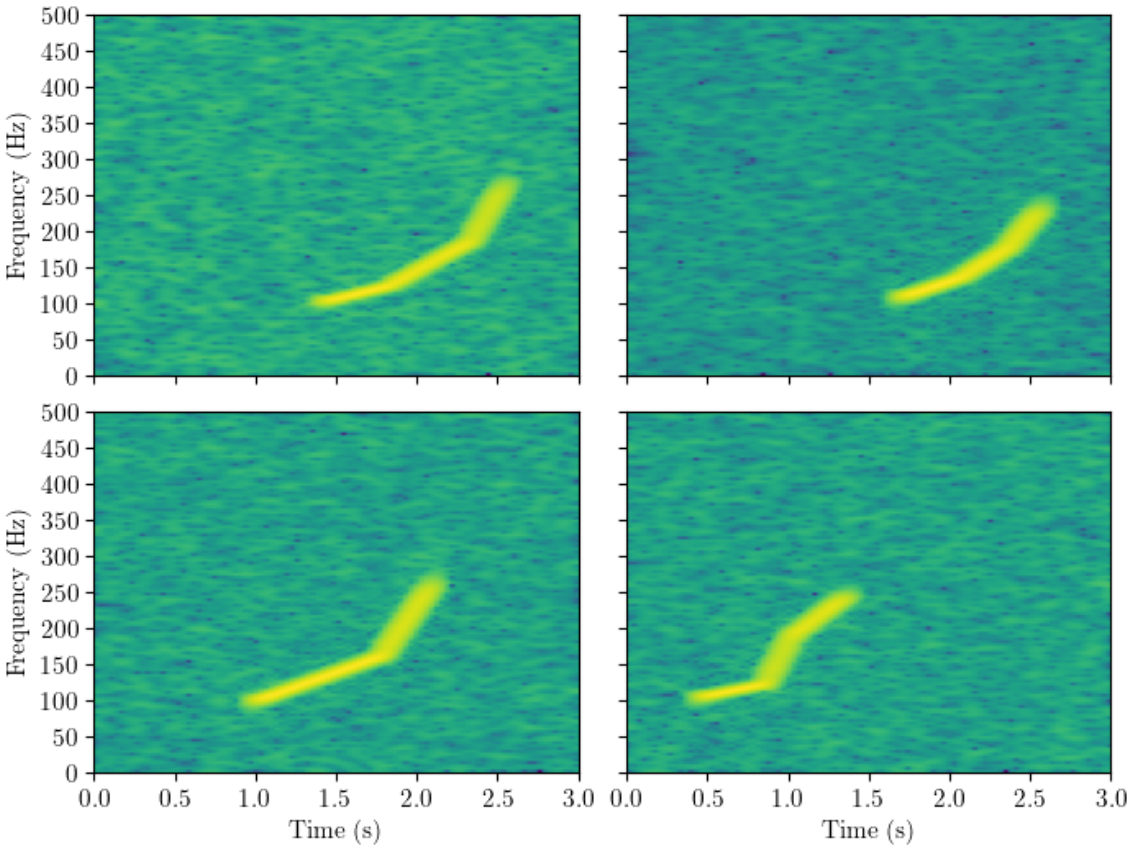

Figure 6.2: A collection of 4 spectrograms displaying different shapes for the synthetic upcalls achieved by the proposed algorithm. For details on how the synthetic NARW upcall samples were generated, see Section 6.1.3

how a detector and classifier will perform under varying conditions. Each underwater environment possesses unique characteristics, such as temperature, salinity and bathymetry, which play a crucial role in shaping the acoustic environment. Moreover, these environments are subject to various influencing factors, including underwater currents, different sources of transient sounds/noise, geological features, and different ambient sounds. Even neighboring locations can experience significant variations in propagation conditions. This variability poses a substantial challenge to a DNN ability to generalize and adapt to different underwater settings.

In this thesis, we employ acoustic propagation modeling to simulate the transmission of "clean" synthetic upcalls within the target environment. These upcalls represent signals that have not been affected by any propagation-related distortions. This approach is motivated by the results obtained in Chapter 5, which highlighted

the importance of adapting a DNN to the target underwater environment for detecting NARW upcalls. By propagating the synthetic signals, we provide the DNN with valuable information concerning the acoustic environment, rather than merely focusing on the properties of the upcalls themselves. Figure 6.3 presents a synthetic NARW upcall propagated through different virtual underwater environments. This figure illustrates the complex interaction of the upcall with various environmental factors, such as water depth, seafloor composition, and sound speed profiles, which ultimately shape the characteristics of the received signal.

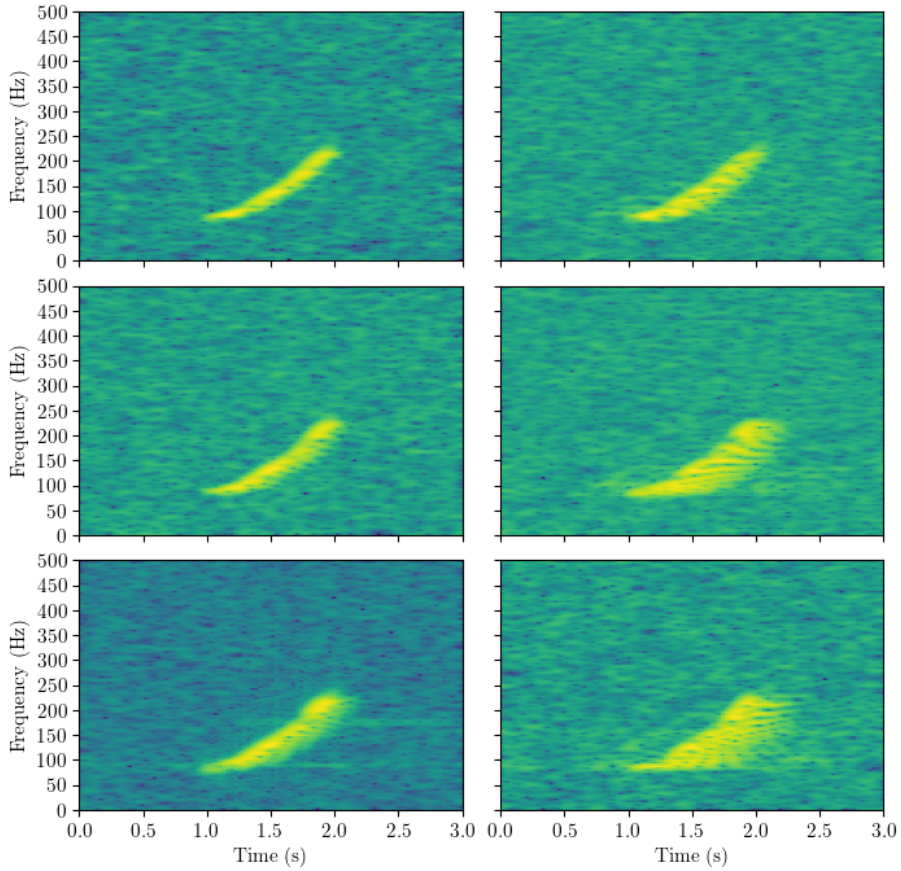

Figure 6.3: A side-by-side comparison of synthetic NARW upcalls before and after acoustic propagation modelling through various simulated underwater environments. The spectrograms on the left displays the synthetic upcalls propagated through a sandy bottom at a depth of 70 m while those on the right corresponds to synthetic upcalls propagated through a gravel seafloor at a depth of 120 m. The top, middle, and bottom rows represent propagation distances of $2,500$ m, $7,500$ m, and $10,000$ m, respectively. Details on the acoustic propagation modelling can be found in Section 6.1.4. White noise has been added to the background to simulate environmental noise.

The propagation modeling conducted in this thesis also assumes an unperturbed environment, devoid of influencing factors caused by underwater currents or abrupt changes in the acoustic properties of the environment caused by temporary external factors. Although a comprehensive modeling of the Gulf of St. Lawrence's acoustic conditions would be an interesting research subject, the main goal of this chapter is to utilize synthetic data for training a DNN without access to annotations, and not to create the best possible acoustic propagation modelling for the Gulf of St. Lawrence.

This thesis utilized the Kadlu[1] Python package implementation of sound propagation, which is based on the methods described by Jensen *et al.* [58]. The Kadlu package uses a parabolic equation (PE) solver (Chapter 6 of Jensen *et al.*, refer to the Kadlu technical note in Appendix A for more details) to compute the transmission loss as a function of frequency, and from this computation, the response function in the time domain is derived through Fourier transform (Section 8.2 in Jensen *et al.*) through the Kadlu-echo package implementation[2]. Finally, this response function is convolved with the synthetic signal to simulate the various propagation effects in the underwater environment. These effects include reverberations and echoes, which are caused by reflections off the water's surface and the seafloor, as well as refraction due to variations in sound speed with depth. Listing 6.1 provides the Python code used for generating the response functions. Propagation modelling is performed on the synthetic upcalls based on the recording site characteristics. Water depth and receiver depth data were obtained from the dataset description provided in Section 3.2 and in Kirsebom *et al.* [65]. While a comprehensive mapping of the Gulf of St. Lawrence's bathymetry should be considered for studies focused on a more in-depth analysis of the propagation effects, we simplify the process by using the reported depth values of the deployments for the dataset.

```
from echo import echo

water_depth = [75, 85, 95, 105, 115, 125] # Defining the bathymetry
    values
seafloor_name = ['sand', 'gravel'] # Defining the lythology data

# loading the Sound speed profile
```

[1]The Kadlu package is hosted at: https://git-dev.cs.dal.ca/meridian/kadlu
[2]The Kadlu-echo package is hosted at: https://git-dev.cs.dal.ca/meridian/kadlu_echo

```
mat = scipy.io.loadmat("Rho_SV_over_depth_ext_2019-09-19_12_NS_1.mat")

# Converting the ssp to the format required by the response functions
ssp = (mat['SV'], mat['depth'])
c = ssp[0].flatten()
z = ssp[1].flatten()
ssp = (c, z)

for floor in seafloor_name:
    seafloor = echo.seafloor_dict[floor] # Retrieving lythology data
    for depth in water_depth:
        # Defining the source depth
        source_depth = np.linspace(5, depth-5, 6, dtype=int)

        # Compute Frequency Response
        mag, phase, axes = echo.compute_freq_response(load_bathymetry=depth,
            seafloor=seafloor, ssp=ssp, source_depth=source_depth, receiver_depth=depth-5,
            receiver_distance=[500, 1000, 2500, 5000, 7500, 10000], freq_max=int(sr/2),
            plot_freq=[200], plot_dir='.')

        # Save frequency Response
        filename = 'freq_response_' + str(floor) + '_' + str(depth) + '.csv'
        echo.save_freq_response('./response_functions/' + filename, mag, phase, axes)
        freq, mag, phase, tag = echo.load_freq_response('./response_functions/' + filename)

        # Derive the response function in time from teh frequency response
        response = []
        for idx, _ in enumerate(mag):
            time, resp = echo.freq2time(freq=freq, mag=mag[idx], phase=phase[idx], window_len=duration)
            response.append(resp)

```

```
38            # Save the time response to convolve with the signal
39            filename = 'time_response_' + str(floor) + '_' + str(depth)
    + '.csv'
40            echo.save_time_response('./response_functions/' + filename,
    time, response)
```

Listing 6.1: The code computes frequency and time response functions for underwater acoustic signals across various water depths and seafloor types, using a given sound speed profile. Results are saved in '.csv' files.

Hydrophones for this dataset were deployed at depths ranging from approximately ~ 75 m to ~ 125 m and positioned about ~ 5 m above the seafloor. Bearing this in mind, we varied the water depth value from 75 m to 125 m and the receiver depth value from 70 m to 120 m in increments of 10 to encompass the deployment depth range. For the horizontal distance between the source (a vocalizing NARW) and the recorder, six distance values were selected, ranging from 500 m to 10000 m. The source depth were selected as 6 equally spaced values between 5 m and the receiver depth. Lithology data for the seafloor were extracted from Loring and Nota (1973) [80] and transformed into acoustic properties using the approximations presented in Table 1.3 of Jensen *et al.* (2011) [58]. Two seafloor compositions were considered: a sandy seafloor with a sound speed of 1650 m/s, a density of 1.9 g/cm^3, and an attenuation of 0.8 dB/λ_p; and a gravel bottom with a sound speed of 1800 m/s, a density of 2.0 g/cm^3, and an attenuation of 0.6 dB/λ_p. Finally, the sound speed profile was computed using the Intergovernmental Oceanographic Commission standard TEOS-10 [87]. Since the water-column properties from the deployments was unavailable, we utilized measurements collected during a separate glider deployment in the Gulf of St. Lawrence in 2019, between September and October. Although employing multiple sound speed profiles throughout the year would offer a more comprehensive representation of the Gulf's acoustic properties, using a single profile for propagating the synthetic data serves as an acceptable approximation for this task. The sound speed profile used in this thesis can be visualized in Figure 6.4. All model parameters used for the acoustic propagation modelling are summarized in Table 6.1.4.

It is important to highlight that initially, we set the horizontal distance between the vocalizing NARW and the recorder to range from 50 m to 1,000 m. However, by extending this propagation distance to 10,000 m—a value well within the typical

vocalization range of a NARW—we observed significant improvements in the performance of the resulting model trained on this data. Two potential explanations can be offered for this enhancement. First, the additional propagation distance generated synthetic samples that more closely resembled natural upcalls, thereby improving the model's ability to detect them. Second, the increased range of distance values contributed to greater diversity and intrinsic variability in the synthetic dataset, attributes that generally contribute to the efficacy of DL models.

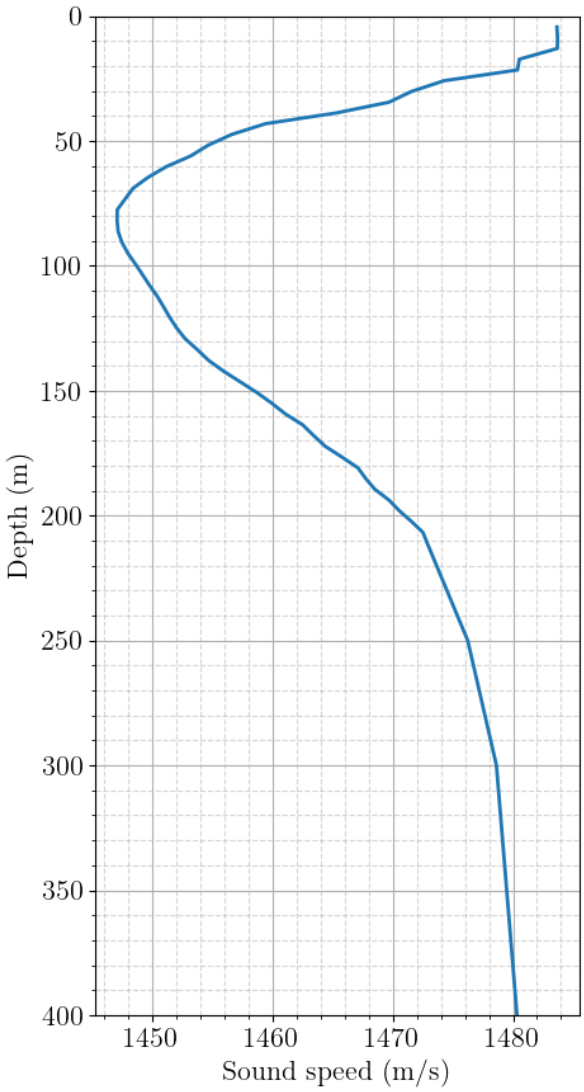

Figure 6.4: Sound speed profile used to calculate the transmission loss for the acoustic propagation modelling of the synthetic signals.

As illustrated in Figure 6.4, the sound speed profile exhibits a variation of approximately 40 m/s within the depths of interest. This represents a significant change, particularly within such a small depth range, and when compared to other similar studies, such as the sound speed profiles presented in Figures 4.5 and 4.7 in Binder [12]. This further underscores the challenge in accurately characterizing the target environment. However, it is yet to be determined whether such variations in sound speed have a substantial impact on the performance of a DNN compared to using a constant value, or if there are other contributing factors that prove to be more influential.

Parameter	**Value**
Water Depth	[75, 85, 95, 105, 115, 125]
Receiver Depth	[70, 80, 90, 100, 110, 120]
Horizontal Distance	[500, 1000, 2500, 5000, 7500, 10000]
Source Depth	[5..Receiver Depth] (6 equally spaced values)
Seafloor Composition (Sandy)	Sandy, Gravel
Sound Speed Profile	Figure 6.4

Table 6.3: Parameters employed for the acoustic propagation modeling utilized in this study, including water depth, receiver depth, horizontal distance, source depth, seafloor composition, and sound speed profile.

6.1.5 Embedding Into Ambient Noise

While the steps described thus far can create synthetic samples of NARW vocalizations and propagate them through a virtual environment representing the Gulf of St. Lawrence, they still lack the contextual information present in the background environment of real samples. This contextual information, such as various sources of transient sounds/noise and different ambient sounds, can help a DNN distinguish between natural NARW upcalls and the environmental soundscape. To address this, we embed the synthetic samples in real background samples randomly extracted from the database. The resulting sample will contain both the synthetic upcall information and the information from the soundscape, providing the DNN with a more realistic and complex training dataset.

This approach can be easily applied to any type of synthetic vocalization, as we do not need to verify if the real background segment extracted from the environment

contains a real vocalization. Due to the sparse nature of marine mammal vocalizations, we can safely assume it does not. While in theory, this process could end up embedding a synthetic signal with a real vocalization, the odds are so low that it can be safely disregarded, at least for the context of training a DNN for the detection and classification of marine mammals.

To embed the synthetic signal, we employ the mixup augmentation technique described in Section 4.3.2, which involves blending the synthetic signal with a background segment using a weighted average. It is worth highlighting that we are superimposing the waveforms, not the spectrograms. This process not only combines the synthetic signal with a background segment but also introduces a level of variability into the resulting sample, further enhancing the robustness of the DNN's learning process. Figure 6.5 displays a synthetic NARW upcall embedded in a randomly sampled background segment extracted from the GSL continuous dataset, illustrating how the mixup algorithm effectively incorporates the synthetic vocalization into the environmental soundscape while preserving the characteristics of both components. The resulting composite signal aids the DNN to better generalize and adapt to various acoustic conditions, ultimately improving its performance in detecting and classifying marine mammal vocalizations.

When embedding the synthetic upcall in the background segment, it is important to consider difference in SNR between the embedded synthetic samples relative to the ambient background noise and the natural upcalls relative to the background. If the embedded sample has an excessively high SNR, it can introduce artifacts from the spectrogram computation of the synthetic signal generation into the embedded spectrogram visualization, as seen in Figure 6.5. Moreover, the DNN may learn to search for this high SNR pattern, incorrectly classifying natural upcall samples as negative. Conversely, if the embedded sample has too low an SNR, the synthetic signal features may be overwhelmed by the background noise, rendering it indistinguishable. In Figure 6.6, we compare the embedded synthetic upcall sample from Figure 6.5 to a natural upcall sample. We can observe that the artifacts in the embedded synthetic sample caused by spectral leakage are not present in the natural upcall. Additionally, we can observe that the natural upcall exhibits a lower intensity and seamlessly integrates with the background environment. It is worth noting that spectral leakage

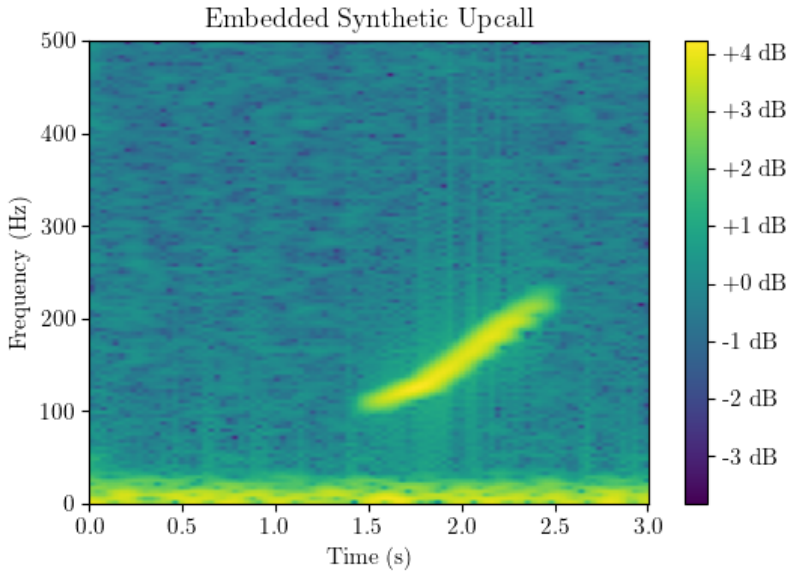

Figure 6.5: A spectrogram showcasing a synthetic NARW upcall embedded into a randomly sampled background segment extracted from the GSL continuous dataset using the mixup augmentation technique. Details on the embedding process can be seen in Section 6.1.5

(side-lobes) can be addressed using various windowing functions; however, to maintain consistency with the spectrogram computation performed throughout this work, artifacts are addressed by adjusting the SNR of the embedded synthetic upcalls.

6.1.6 SNR Adjustment

To adjust the SNR of an embedded sample to resemble natural segments, we manipulate the mixing weights α and β of the mixup method when incorporating the synthetic upcall into the background environment. Our goal is to determine appropriate α and β values that ensure the synthetic signal remains audible and distinguishable within the background segment while maintaining a similar SNR to the target dataset. Given a synthetic signal S and a real background segment B an embedded sample x can be constructed by:

$$x = \alpha B + \beta S \tag{6.4}$$

To find an appropriate β weight, we first compute the standard deviation of each recording in the target dataset and use the mean value σ_r as a reference. This

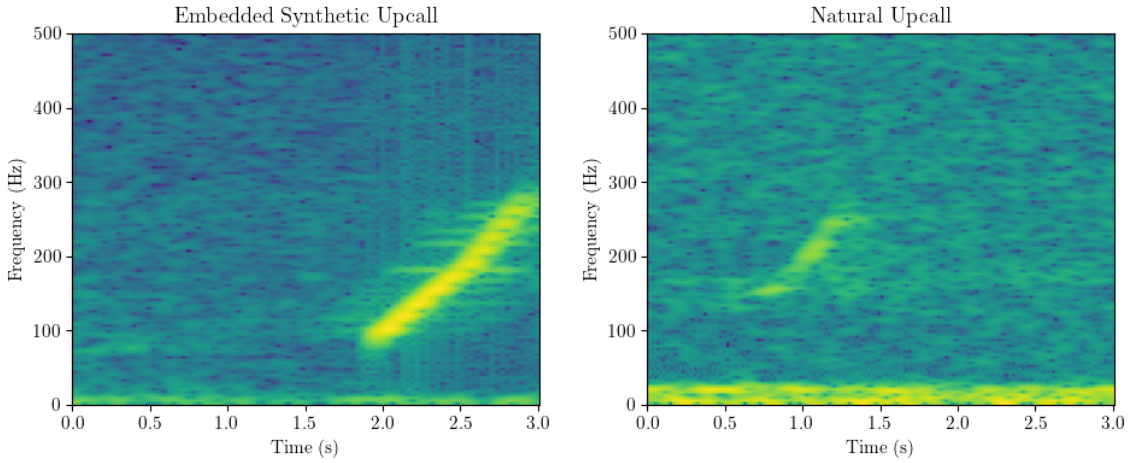

Figure 6.6: A comparison of an embedded synthetic upcall sample (left) and a natural upcall sample (right) from the GSL dataset. The figure illustrates the differences in intensity in the synthetic upcall before SNR adjustment. The natural upcall seamlessly integrates with the background environment, highlighting the importance of proper SNR adjustment for synthetic upcalls to ensure realistic training data for the DNN.

computation is performed on the waveforms and not the spectrograms. We also calculate the standard deviation of each synthetically generated upcall and derive a mean value σ_s. We can then determine β as the scaling factor between σ_s and σ_r, adjusted by a random value. Thus, we have:

$$\beta = \frac{\sigma_s}{\sigma_r} + h \tag{6.5}$$

where h is a random value within the range $H = [0.0, -0.07]$. Empirically, we found that adjusting β to cover a wider range of values generated a more realistic looking range of signals, varying from strong to barely distinguishable from the background environment. The range H can be adapted to create a more suitable set of mixing weights that properly target an specific dataset's unique properties, such as the overall noise level. By tuning the range H accordingly, we can ensure that the embedded synthetic signals better resemble the natural upcalls in the target dataset, resulting in a more effective training process for the DNN. With β determined, we can assign α as $1 - \beta$. Balancing α and β in this manner allows the DNN to learn to detect upcalls amidst the same level of noise present in real-world scenarios. Figure 6.7

displays an embedded synthetic upcall after applying the mixing weights. The intensity of the signal now closely resembles that of a natural upcall, and in addition, any artifact from the spectrogram computation of the synthetic upcall is effectively masked. It is worth noting that, in theory, the upcall SNR relative to the background should be correlated with the propagation distance, with calls becoming fainter as the hydrophone is further away from the whale. However, we opted for a simpler approach to adjust the SNR, as we believed it would be sufficient for generating useful training data. Furthermore, the variability in sound source levels and ambient noise levels means that the correlation between received level and distance won't be entirely precise. Another limitation is the way β is calculated: it is designed to match the SNR of the entire dataset as a whole rather than accounting for the variability in SNR across individual samples. As a result, since we are using a range of values to approximate the generations to the dataset's average SNR, some synthetic calls may be anomalously low or high in relation to the background noise compared to real samples. However, it is worth noting that these slightly unnatural generations could actually be beneficial for training a DNN as they encourage the network to learn a more generalized representation of the data space.

To validate that our approach produces embedded signals with a SNR comparable to those in the GSL dataset, we compute the SNR of the embedded synthetic upcalls and compare them to the SNR of real upcalls found in the dataset. It is important to note, however, that this computation is not a requirement for applying the methodology, especially in real-world scenarios where real upcall annotations may not be available. This comparison serves primarily to demonstrate the effectiveness of our technique in generating synthetic signals with similar SNR to the real upcalls in the dataset.

Various methods exist for calculating SNR, ranging from complex, accurate approaches to simpler ones that yield rough approximations for each sample. For example, Binder *et al.* [14] computes the SNR of denoised spectrograms through:

$$SNR = 10\log_{10}(\frac{\sigma^2_{s+n}}{\sigma^2_n}) \tag{6.6}$$

where σ^2_{s+n} is the combined signal and noise variance, and σ^2_n is the noise variance. Given that in their work, each spectrogram spans a couple of seconds, this formula

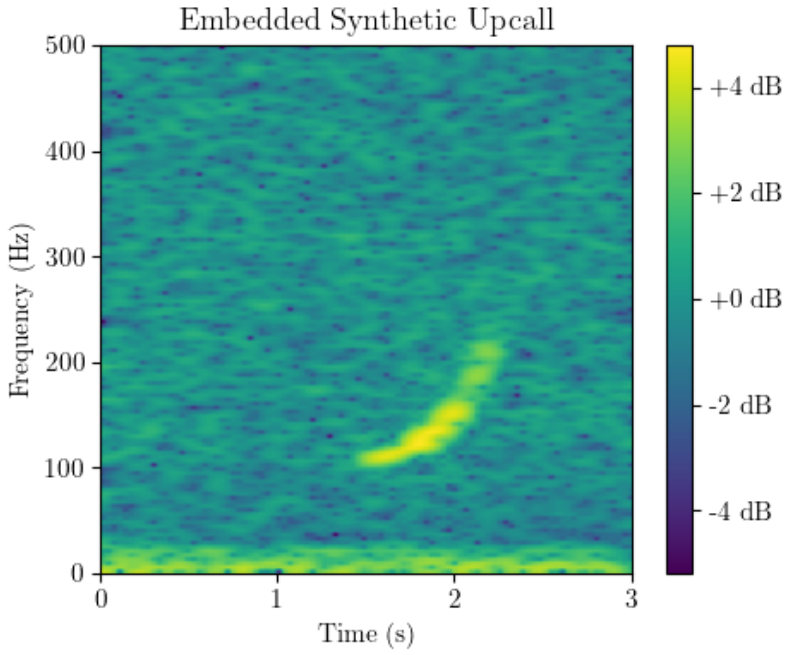

Figure 6.7: A spectrogram showcasing a synthetic NARW upcall embedded into a randomly sampled background segment extracted from the GSL continuous dataset using the mixup augmentation technique. Here, the mixing weights were applied to seamlessly blend the synthetic upcall with the background noise, creating a more realistic representation. For details on the embedding process, refer to Section 6.1.5 and Section 6.1.6

provides a useful approximation. Spectrograms with high SNRs are subsequently selected for further analysis, particularly for studying the impact of applying propagation models.

In this thesis, we adopt a more sophisticated method to estimate the SNR value of each sample, whether positive or negative. This approach follows a heuristic algorithm outlined in [65]. The procedure consists of the following steps:

1. **Denoising the Spectrogram:** A denoised spectrogram X_d was generated by first subtracting the median value of each time slice to minimize broadband, impulsive noise. Next, the median value of each frequency slice was subtracted to eliminate tonal noise.

2. **Locating the Upcall Mid-Point:** The mid-point of the upcall is identified by sliding a one-second window across X_d and finding the maximum pixel value in time within the frequency range of 80–200 Hz as given by:

$$\text{sum}_t(\max_f(X_d)) + \text{sum}_f(\max_t(X_d)) \tag{6.7}$$

where t and f denote the time and frequency axes, respectively.

3. **Creating a Trace:** A trace is generated by connecting the pixels in X_d that have the maximum value within each time slice.

4. **Computing Median Value Along Trace:** The median value of the original spectrogram X is calculated along this trace. To accommodate the finite "width" of the upcall, three pixels immediately above and below the trace were also included.

5. **Final SNR Estimation:** The median values of X in the 0.5-second adjacent windows are computed for the frequency interval of 80–200 Hz and then subtracted from the original value.

Before adjustment, the average SNR of the synthetic dataset was 33.91 dB with a range spanning from -10.37 dB to 67.14 dB. In contrast, the average SNR of the real upcalls was 10.14 dB, with a range extending from -0.68 dB to 27.85 dB. After adjusting the synthetic samples, their average SNR value became closer to the real upcalls, reaching 19.7 dB, and spanning a range from 2.71 dB to 39.88 dB. In Section 6.2, we evaluate and compare the performance of models trained on a dataset where synthetic signals were embedded but not adjusted in terms of their SNR, and a dataset where synthetic signals were embedded and adjusted. It is important to clarify that the objective is not to perfectly match the SNR between synthetic and real upcalls. Instead, the aim is to verify that our methodology generates synthetic signals whose SNR characteristics are not exaggeratedly different from those of real upcalls.

6.1.7 Experimental Protocol

In this section, we present a series of experiments designed to evaluate the performance of our proposed methodology for generating synthetic NARW upcall datasets and subsequently training a DNN on this data for the classification and detection of real

NARW upcalls. We utilized the ResNet-18 architecture described in Section 3.4 for all experiments. A batch size of 128 samples and a learning rate of 0.001 were employed using the ADAM optimization algorithm. For all experiments, the models were evaluated on the 30-minutes recordings of the GSL dataset. All experiments were conducted on a workstation with a NVIDIA TITAN V GPU with 12 GB of memory using the NARW detection tool described in Section 5.1.4 for the training and testing. For all experiments, we used a batch size of 128 samples and learning rate of 0.001 for 20 epochs using the ADAM optimization algorithm. Empirically, we found that the model was quickly converging when trained on the synthetic data and a low number of training epochs was enough to evaluate performance. In particular, we observed that the models converged more rapidly on synthetic upcalls compared to natural upcalls for datasets of similar size.

Our goal is to assess whether the synthetic data could yield results comparable to those obtained from real upcall samples when training a DNN for the detection and classification of real NARW upcalls. While we expect that a model trained on real data would still outperform a model trained only on any form of synthetic data, we aim to evaluate the extent of this difference and whether the synthetic data can still provide sufficient accuracy to be usable by bioacousticians. For this purpose, we established a baseline performance by training a ResNet model on the GSL_B clips dataset, which contains real upcall samples. Next, we trained a ResNet model on a synthetic dataset comprising synthetic upcall samples and randomly sampled background samples. We generated $N = 5000$ synthetic upcall samples using the methodology described in previous sections. For embedding the synthetic upcall samples, random background samples were extracted from the 30-minute GSL dataset as outlined in Section 6.1.2. An equal number of negative samples were randomly extracted from the same dataset. In total, the ResNet was trained on 10,000 samples, including 5,000 synthetic NARW upcalls and 5,000 real background samples. With both the baseline performance and the performance of the model trained on synthetic data, we assess whether the latter achieved a level of performance sufficient to be a useful tool for bioacousticians analyzing acoustic data. By comparing the results, we can determine the effectiveness of synthetic data in training the model and its potential for practical application in real-world scenarios.

To investigate the impact of SNR on the performance of trained models, we conduct an additional evaluation of both the baseline and synthetic models, focusing on how they perform as a function of SNR. Specifically, we examine the effects of discarding samples with SNR values below a pre-defined minimum threshold, SNR_{min}, when running the models on the 30-minute GSL test dataset. This allows us to understand how variations in SNR could potentially influence the model's accuracy and its impact on choosing an appropriate recall and precision level.

To assess the contribution of each step in the methodology – synthetic upcall generation, acoustic propagation modeling, embedding synthetic signals into ambient noise, and adjusting the SNR of the embedded samples – to the overall performance of the ResNet model, we trained 5 additional models, isolating each operation and testing each combination. This process enabled us to examine how each component influences the performance of a DNN trained on a synthetic database as the methodology increases in complexity. Table 6.4 display the all the combinations executed. For each experiment, the same number of synthetic samples were generated and background noise was extracted, as described earlier in the text.

Model Name	**Synthetic Upcalls**	**Acoustic Propagation**	**Embedding**	**SNR Adjusting**
S	X			
SP	X	X		
SE	X		X	
SPE	X	X	X	
SE-SNR	X		X	X
SPE-SNR	X	X	X	X

Table 6.4: Combinations of methodology components utilized when training each model. Model names indicate the components used in their training, where **S** represents Synthetic Upcalls, **P** represents Acoustic Propagation, **E** represents Embedding, and **SNR** represents SNR Adjusting.

In addition to the experiments outlined above, we conducted a final experiment to observe the improvement in performance as we incrementally introduced real data into the synthetic models. To this end, we utilized the TL methodology described in Chapter 5 to adapt a model that was initially trained only on synthetic data with incremental amounts of real data. For the model trained on synthetic data, we selected the SPE-SNR model and incrementally incorporated real upcall samples

from the GSL_B clips dataset. For each adaptation procedure, a new model was generated by progressively incorporating real data samples into the training process. Starting with an initial batch of 300 upcall samples from the GSL_B clips dataset, we increased the number of samples in increments of 200 until reaching a total of 2,374 upcall samples. At each training iteration, an equal number of negative (background) samples were also fed to the DNN, ensuring a balanced training set. This approach allowed us to observe the impact of gradually adding real data to the model's training and evaluate its effectiveness in enhancing the model's performance.

This experiment aimed to provide insight into how this type of synthetic model would be used in a real-world setting. The process involves using the model trained on synthetic data to create an initial batch of real detections to be analyzed, which can then be fed back into the model to improve its performance. This iterative approach allows for continuous enhancement of the model's performance as more real NARW upcalls are detected, demonstrating the practical utility of models trained on synthetic data for assisting ocean researchers in detecting marine mammal vocalizations as well as highlighting the potential for ongoing adaptation and refinement.

We employ the same evaluation metrics of precision, recall, and FPR per hour as described in Section 3.5 to assess the performance of the models. To account for variability due to random initializations, we repeat the training processes 10 times with different initial network weights and biases. The results are then averaged to establish the mean performance.

6.2 Results

Figure 6.8 and Figure 6.9 summarize the performance of the SPE-SNR model on the GSL 30-minutes recordings test dataset. In Figure 6.8 we show precision against recall, while in Figure 6.9 we display the FPR against recall. The dashed lines indicate the baseline performance while the solid lines indicate the performance of the SPE-SNR model that was trained on data generated with all the steps outlined in the previous sections.

For recall levels below ~ 0.83, the ResNet-18 model trained on synthetic data outperforms the baseline model, maintaining a precision greater than 0.80 with a standard deviation of $\sim 4\%$. On the other hand, for recall values above ~ 0.83, the

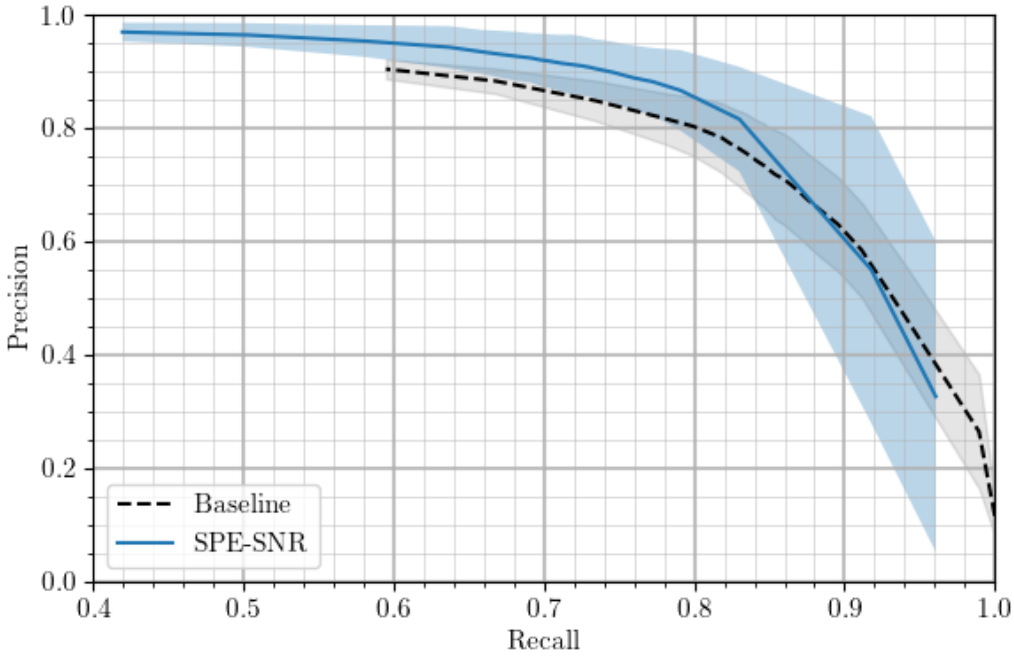

Figure 6.8: Precision vs. recall on the GSL_B 30-minutes full-length recordings. Results obtained with the SPE-SNR model are shown in blue while results obtained with the baseline model are shown in black. The dashed lines also indicate the baseline model's performance. The confidence bands denote the range within one standard deviation of the mean.

SPE-SNR model experiences a sharp drop in precision, to 0.35, while retrieving 93% of the calls. This decline is also accompanied by an increase in the standard deviation, as evidenced by the expanded shaded blue region in the plot. In contrast, the baseline model, which was trained on real data from the GSL_clips dataset, sustains a higher performance, achieving a precision of ~ 0.5 at similar recall levels.

A similar pattern can be observed int the FPR per hour for most levels of recall. For recall levels below 0.82, the SPE-SNR model trained on synthetic data generates fewer than 10 FP per hour. Furthermore, at these recall levels, the SPE-SNR model yields a slightly lower FPR when compared to the baseline model. Conversely, for recall levels higher than 0.82, the baseline model demonstrates a substantially lower FPR than the SPE-SNR model. This sharp rise in the FPR of the SPE-SNR model, which can be attributed to its diminished precision, is evident in the accompanying figure.

A similar pattern can be observed for the FPR per hour for most levels of recall.

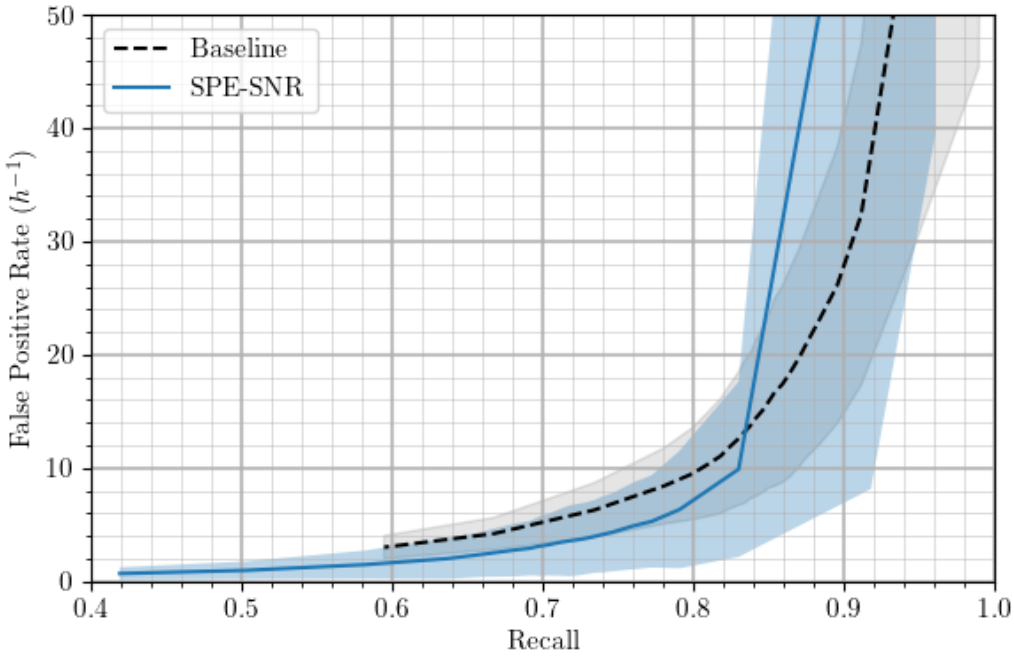

Figure 6.9: Recall vs FPR on the GSL_B 30-minutes full-length recordings. Results obtained with the SPE-SNR model are shown in blue while results obtained with the baseline model are shown in black. The dashed lines also indicate the baseline model's performance. The confidence bands denote the range within one standard deviation of the mean

The baseline model trained on real data has higher recall levels for the same FPR when the recall is greater than 0.6. Conversely, for recall levels lower than 0.5, the SPE-SNR model achieved fewer than 4 average false positives per hour, lower than what the baseline achieved for any given recall.

The SPE-SNR model exhibited a much higher overall standard deviation than the baseline model, as inferred from the confidence intervals in Figure 6.9 and Figure 6.8. This was the case for all precision and FPR per hour values for any given recall. This suggests that the five training iterations produced models with varying levels of performance for both recall and precision. For instance, with a default detection threshold of 0.5, one training iteration produced a model with a recall of ~ 0.84 and precision of ~ 0.72, while another generated a model with a recall of ~ 0.53 and precision of ~ 0.95. Notably, a precision of ~ 0.95 is higher than any of the baseline models, achieving a particularly low FPR per hour of fewer than 2.

In Figure 6.10, we evaluate the performance of the SPE-SNR and baseline models

as functions of SNR, with all results obtained using a detection threshold of 0.5. Both models exhibit similar trends with a gradual performance improvement in precision and recall as the SNR threshold, SNR_{min}, increases. Notably, precision increases for SNR values beyond 0, while recall rises starting from an SNR value of 5. The recall for both models, along with the precision of the baseline model, exhibits a sharp initial increase. This concurrent improvement in precision and recall contributes to a decrease in the FPR per hour when increasing the SNR_{min} threshold. For instance, at an SNR_{min} threshold of 10 dB, the SPE-SNR model identifies $\sim 85\%$ of the upcalls with precision ~ 0.95 and yields fewer than 2 false positives per hour.

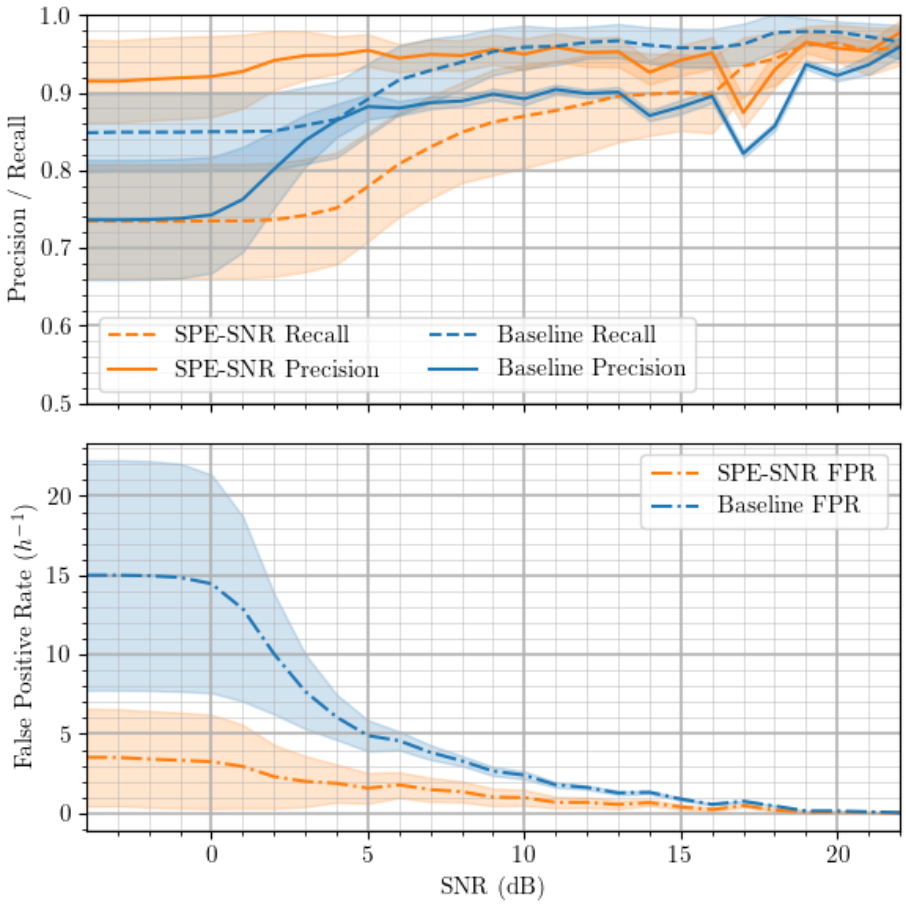

Figure 6.10: Precision, recall and FPR vs SNR on the GSL_B 30-minutes full-length recordings. The top sub-plot illustrates the mean recall (dashed lines) and precision (solid lines) for both the synthetic models (orange) and the baseline models (blue). The bottom sub-plot shows the FPR for both models. Error regions, shown in lighter shades around each line, represent one standard deviation from the mean. The experiments were tan with a detection threshold of 0.5

In Table 6.5, we summarize the performance of various models trained with different combinations of methodology components, with all results obtained using a detection threshold of 0.5. The baseline performance is also displayed for comparison. Initially, we observe that the simpler approaches using only one or two components (models S, SP and SE), indicated by dashes in the table, failed to distinguish real upcalls from the background environment. These models classified all test set samples as negatives.

Conversely, the SPE model, which incorporated synthetic upcalls, acoustic propagation, and embedding but did not use SNR adjusting, was the first combination where the trained model on synthetic data demonstrated some discrimination capabilities. However, the SPE model only retrieved 3% ($\pm$0.01) of the upcalls from the test recordings. The SE-SNR model, which incorporated synthetic upcalls, embedding, and SNR adjusting without acoustic propagation, achieved the highest precision of 0.96($\pm$0.01) and a recall of 0.37($\pm$0.11), resulting in a very low FPR per hour of 0.57($\pm$0.27). The SPE-SNR model, which combined all methodology components, achieved a precision of 0.91($\pm$0.05) and the highest recall among synthetic models at 0.71($\pm$0.08), yielding an FPR per hour of 3.57($\pm$3.13). This combination of components provided a more balanced performance compared to other models. For comparison, the baseline model, trained on real data, achieved a precision of 0.73($\pm$0.07), a recall of 0.85($\pm$0.06), and an FPR of 15.13($\pm$7.30). In general, we observe that the baseline model achieves a substantially higher recall than models trained on synthetic data, but at the cost of lower precision.

We also investigated the effect of incrementally introducing real data into the synthetic models through TL. These findings are presented in Figure 6.11, where we display precision, recall, and the FPR per hour as a function of the number of additional real samples incorporated into the model. Generally, we found that the introduction of even a small amount of real data (600 samples) was crucial in substantially increasing the model's recall from $\sim$ 0.57 to $\sim$ 0.8. The addition of more real samples did not lead to further increases in recall. This increase in recall was accompanied by a slight increase in precision from 0.85 to 0.95 with only 300 real upcall samples. As we introduced more real data, precision produced a downward trend until 900 real upcall samples, at which point precision stabilized at $\sim$ 0.9.

Model Name	Precision	Recall	FPR (h)
Baseline	0.73(±0.07)	0.85(±0.06)	15.13(±7.30)
S	-	-	-
SP	-	-	-
SE	-	-	-
SPE	0.85(±0.12)	0.03(±0.01)	0.20(±0.18)
SE-SNR	0.96(±0.01)	0.37(±0.11)	0.57(±0.27)
SPE-SNR	0.91(±0.05)	0.71(±0.08)	3.57(±3.13)

Table 6.5: Combinations of methodology components utilized when training each model. Model names indicate the components used in their training, where **S** represents Synthetic Upcalls, **P** represents Acoustic Propagation, **E** represents Embedding, and **SNR** represents SNR Adjusting. All results were obtained with a detection threshold of 0.5. A dash indicates that the model was not able to produce useful results, and classified everything as negative. The standard deviation is given in the parentheses.

Notably, the substantial increase in recall did not generate in a rise in the FPR per hour, with the every model generating less than 5 FP per hour after model adaptation. Moreover, we note that incorporating real data had a large impact in reducing the standard deviation, as indicated by the confidence intervals. We observed that the addition of 900 or more real upcall samples resulted in considerably lower standard deviations in both precision and recall compared to the models trained solely on synthetic data.

6.3 Discussion

Overall, the results demonstrate that the synthetic data generated by our proposed methodology can effectively train a DNN for the classification and detection of real NARW upcalls. This is evidenced by the performance of the SPE-SNR model, which was able to retrieve over 70% of all the real upcalls from the test set, while producing less than 4 FP per hour of recording. In addition, the SPE-SNR outperformed, on average, the baseline model, which was trained on real vocalizations, when retrieving up to $\sim 82\%$ of the vocalizations. This finding is significant, as no annotated samples were used to train the DNN. The success of the synthetic data-driven approach enables analysis efforts to be conducted after an initial training phase, reducing the reliance on time-consuming and labor-intensive manual annotation of real data. Consequently,

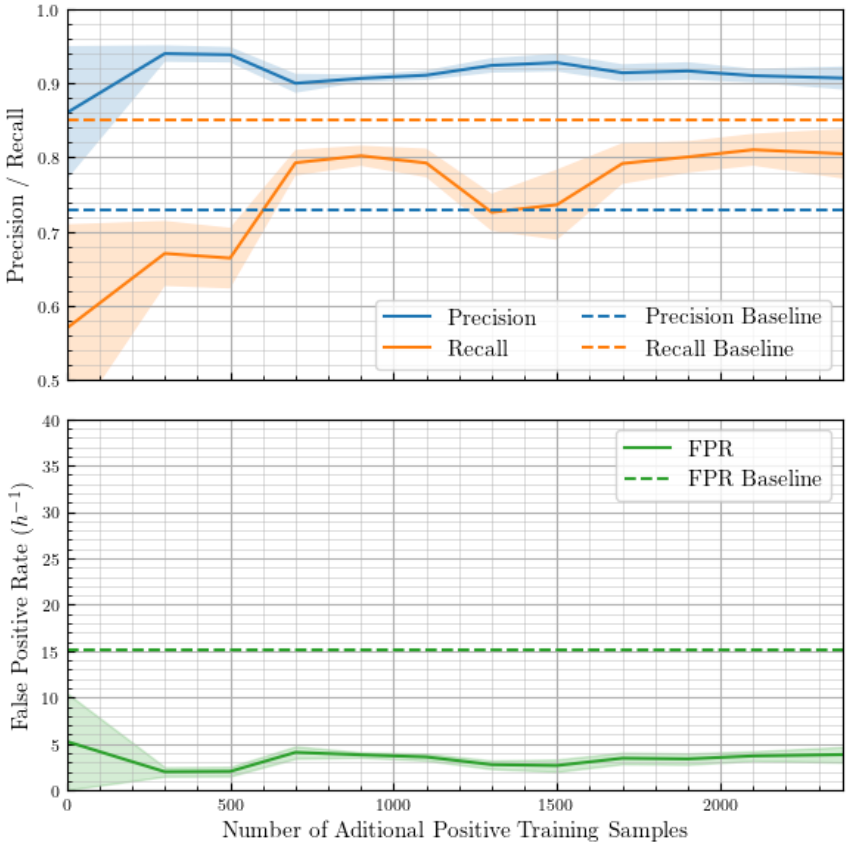

Figure 6.11: Impact of incrementally introducing real positive data samples into the synthetic models through TL. The top graph illustrates the precision and recall as real upcall samples are progressively added, while the bottom graph depicts the FPR per hour with the same incremental additions. Solid lines represent the performance of each model, and dashed straight lines indicate the performance of the baseline model. The confidence bands denote the range within one standard deviation of the mean. An equal number of negative samples were fed to the DNN at each iteration.

this methodology can facilitate more efficient exploration of unprocessed datasets which have not yet been annotated, as well as production of acoustic datasets for training DNNs to detect marine mammal vocalizations.

We have also examined the FN from the SPE-SNR model and found that the majority of the FN were signals that display very different patterns from the stereotypical NARW upcall on which we based our synthetic generation algorithm. This can be seen in Figure 6.12, where we present six spectrograms of real upcall samples extracted from the test recordings that were not detected by the SPE-SNR model trained on synthetic data. From these examples, for instance, we observe that the

upcalls are sometimes only partially present in the spectrogram view, with only a small portion of their frequency range visible and often a lot shorter than the defined one-second-long stereotypical upcall. This suggests that the synthetic models would benefit from a more robust synthetic upcall generation process that covers a wider range of patterns and shape variety observed in real upcall samples. Increasing the variability of the synthetic upcall samples would also have a positive impact in reducing the standard deviation of the synthetic models. More diverse training datasets would better represent the real-world variability in upcall patterns. With a broader range of synthetic upcalls, the model can learn to generalize more effectively, leading to more consistent performance across different iterations. Ultimately however, synthetic upcalls can only provide an approximation of the true variability observed in real upcalls, underscoring the importance of retraining the models once real upcall samples are available.

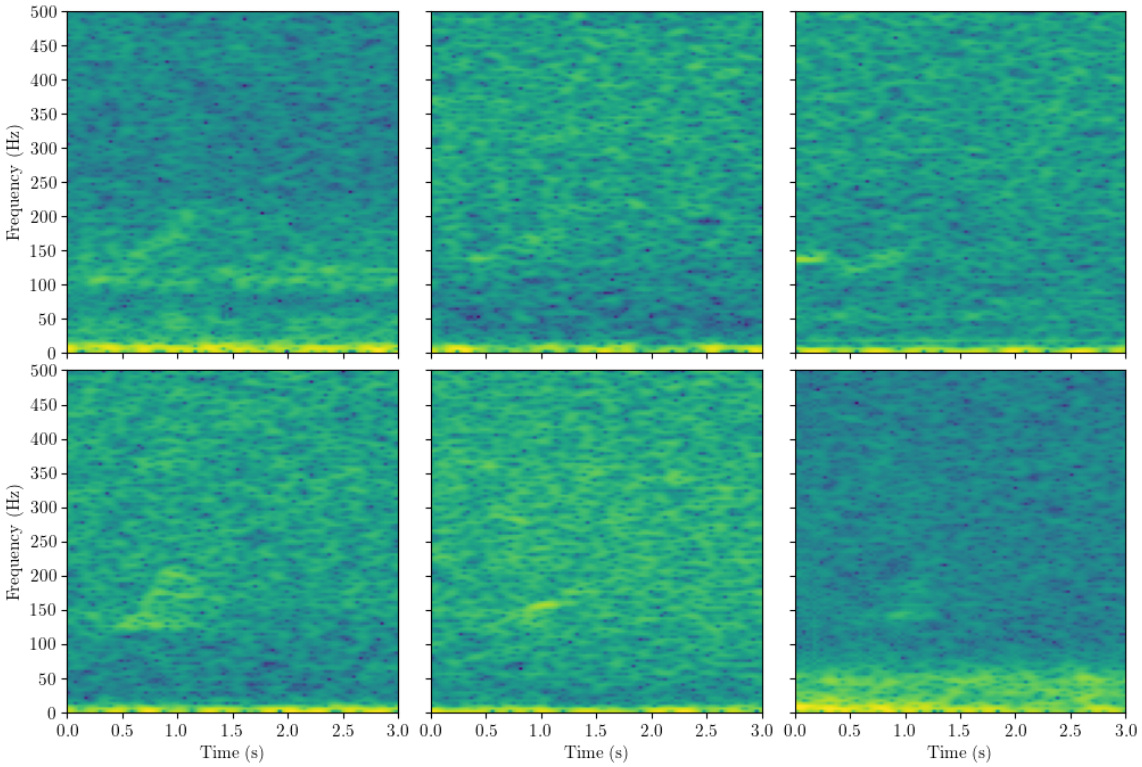

Figure 6.12: A collection of six spectrograms displaying real upcall segments that were not detected by the SPE-SNR synthetic model.

As anticipated, the baseline model trained only on real vocalizations still demonstrated a competitive performance when compared to any of the models trained solely on synthetic upcalls. Additionally, the baseline model exhibited a substantially lower standard deviation compared to the SPE-SNR model across all levels of recall. This difference was particularly pronounced at higher recall levels, where the SPE-SNR model displayed a sharp deacrese in overall performance. This further highlights the importance of using real-world data in training models for the task of detecting and classifying NARW upcalls. By pairing the synthetic models with incrementally more real upcall samples when using model adaptation, we observed considerable improvements in their performance. The addition of even a small number of real samples substantially improved the recall up to ~ 0.8 and reduced the standard deviation of the models, further highlighting the importance of real data in the training process. The increased recall was observed with a slight increase of precision as additional real upcall samples were introduced, achieving a more balanced performance when compared to the purely synthetic models.

We also noted a considerable reduction in the standard deviation as we added real upcall samples. Adapting the synthetic models with real data resulted in a better overall performance when compared to the baseline baseline. However, while the baseline model produce a higher recall than precision (~ 0.85 and ~ 0.73, respectively), the adapted models showed a higher precision than recall (~ 0.87 and ~ 0.8, respectively). This led to a significant reduction in the FPR per hour, to less than 4 for the model trained on synthetic data combined with real-world observations. In comparison baseline model, generated an FPR of 15 per hour. These results further reinforce our findings and are indicative that using models trained only on synthetic data to process large datasets and aid human annotators in identifying and annotating real upcalls to subsequently produce a better model on real data, is a viable approach.

By focusing only on upcalls with an SNR greater than 5.0, the baseline model managed to retrieve over 90% of the NARW upcalls in the test recordings while maintaining an average precision of ~ 0.85 and a substantial reduction in the FPR of around 3x. On the other hand, the SPE-SNR model did not exhibit a substantial reduction in false positives per hour, even at elevated SNR_{min} thresholds. When

the $SNR_{min} = 10$ threshold was set to zero, the SPE-SNR model achieved the best balance between precision and recall, with values of approximately ~ 0.95 and ~ 0.85, respectively, along with a low FPR of 2. These findings are consistent with those in [65], which demonstrated a steady performance enhancement as the SNR_{min} threshold rose. Additionally, a reduced standard deviation was observed for higher SNR values.

Upon examining the contributions of each component in our synthetic NARW upcall generation methodology, we found that the highest-performing synthetic models were those that incorporated all methodological components. This indicates that an effective approach for optimal results relies on the integration of artificially generated synthetic upcalls, acoustic propagation modeling, embedding synthetic upcalls into real background samples, and adjusting the SNR of the embedded synthetic upcalls. Notably, models that combined fewer than three steps of the proposed methodology failed to yield useful results, further emphasizing the importance of producing a diverse array of upcall patterns with varying acoustic properties and environmental noise conditions in order to successfully train models for detecting and classifying real NARW upcalls.

While the proposed approach was able to effectively produce a useful NARW upcall detector, we also identified important limitations. These include: (1) the restricted variation for producing diverse marine mammal vocalization with different properties; (2) the complexity of the acoustic propgation methodological step, which limits usability for users without experience in the field; and (3) the dependency on the understanding and accurate mathematical modelling of the acoustic source properties of marine mammal vocalizations. This methodology's effectiveness is inherently reliant on the ability to synthetic signals that are similar to real vocalizations. However, this process can be far from straightforward. While synthesizing NARW upcalls may be comparatively simpler, vocalizations from other species, such as killer whales, can be highly complex with significant intrinsic variation that cannot be accounted for by propagation effects. Based on these observations, two distinct approaches can be pursued to expand the methodology.

One approach involves delving deeper into the complexity of synthetic data generation, acoustic propagation modeling, and other advanced techniques. By refining

and improving the synthetic marine mammal vocalization generation process, incorporating more accurate propagation models, and optimizing other components of the methodology, we can produce more realistic synthetic vocalizations that approaches what is observed in real-world settings. Consequently, a DNN trained on this data would be better suited to generalize to real-world data.

Alternatively, a more straightforward and simplistic process can also be adopted for generating synthetic vocalizations. This process involves training a DNN on simple shapes that resembles the vocalization of interest. In the case of NARW upcalls, these shapes could include half-parabola-like structures, crescent moons, or other images that mimic an upcall's structure. This methodology was successfully employed by Li *et al.* [75], where the authors generated delphinid whistle-like structures based on contours derived from computer vision images, then superimposed these structures onto spectrograms. Thus, an alternative approach that emphasizes the structure of the vocalizations, while minimizing the complexities introduced by ambient noise or propagation effects (by using denoised spectrograms, for instance) may allow the use of simpler, more abstract structures that resemble real vocalizations without needing to model them in detail. The emphasis on core structural features, rather than intricate variations or background noise, reduces the complexity of the data generation process. The result is a more accessible methodology that capitalizes on the strengths of DNNs to extract meaningful patterns from these simplified representations, while still being capable of accurate marine mammal vocalization detection.

Chapter 7

Conclusion and Future Work

Passive acoustic monitoring (PAM) supported by deep leaning (DL) based detection and classification systems plays an important role in supporting marine mammal research. However, the availability of large fully annotated passive acoustic datasets constitutes a major bottleneck in developing accurate detection and classification systems to identify marine mammals vocalizations. In this work, we presented three (DL) approaches to overcome data scarcity: data augmentation, transfer learning (TL), and artificial generation of synthetic vocalizations, deployed across three distinct phases.

In the first phase (Chapter 4), we developed an approach to improve the performance of a deep neural network (DNN) North Altnatic right whale (NARW) upcall classifier through the use of two data augmentation methods, SpecAugment and Mixup. With a dataset consisting of 3,309 underwater sound clips, about half of which contained a NARW upcall, we trained a Residual Network (ResNet) binary classifier with and without a data augmentation step. By augmenting the training set up to a total of 20,000 samples, we observed a significant improvement in the performance of the DNN. The approach showed a performance increase from 85.9% up to 90.7% for precision and from 89.0% up to 91.3% for recall (Section 4.4). Furthermore, data augmentation in general caused the DNN model to underfit the training data, converting an overfitting problem into an underfitting problem. This suggests that the proposed architecture and training procedure does not fully exploit the potential of the augmented training set, and opens interesting possibilities such as increasing the width and depth of the DNN architecture and increasing the number of training epochs. In addition, to simulate a scenario of severe data scarcity, we undersampled the original dataset and compared the performance of a DNN model before and after data augmentation. We observed a significant improvement in precision and recall when adding the augmented data to the training dataset in different conditions of data scarcity.

In the second phase of the research (Chapter 5), we investigated a transfer learning (TL) approach to address the problem of data scarcity in PAM and enhance a NARW classifier's effectiveness across different underwater acoustic environments. In summary, we utilized TL to adapt ResNet and VGG models that were trained on data from the Gulf of Maine and Gulf of St. Lawrence to detect NARW upcalls in the Emerald Basin. Initial training was conducted using the data from the GOM dataset (9,063 upcalls) and GSL dataset (4,331 upcall), while model adaptation employed the EMB_2015 dataset (232 upcalls). The models were then evaluated on the EMB_2016 recordings (503 upcalls). Overall, we saw a substantial improvement in detector performance after each model was adapted to the new environment, with the ResNet model improving recall from 70% to 85% for a false positive rate of less than 5 per hour (Section 5.2). These results are encouraging as the annotated data used for the adaptation process was considerably less than the amount originally used for training. This allows DNNs to be used in scenarios where there would otherwise be insufficient data to produce a useful detector. We also introduced a command-line interface tool for developing NARW acoustic detectors and classifiers to encourage the adoption of DL in marine mammal research.

In the third phase (Chapter 6), we explored the generation of synthetic NARW upcall vocalizations and their integration with real acoustic properties from the underwater environment to produce training datasets for DNNs. This approach addresses the challenge of data scarcity when no annotated data is available for training DL models. To evaluate our approach, we selected the GSL_B* dataset, which consists of 50 continuous recordings, each 30 minutes long. We developed a methodology for synthetic generation of NARW upcall that involved fours key steps. First, we artificially generated synthetic signals with properties similar to NARW upcalls (Section 6.1.3). Second, we transmitted these synthetic signals through a virtual environment with similar characteristics to the Gulf of St. Lawrence environment to simulate the distortion of an upcall as it propagates through water (Section 6.1.4). Third, we embedded the synthetic upcalls into real background samples randomly extracted from the recordings, incorporating the contextual information present in the real samples' background environment (Section 6.1.5). Lastly, we adjusted the SNR of the embedded synthetic upcalls to resemble that of real upcalls (Section 6.1.6). We then

trained and evaluated DNN classifiers on both purely synthetic and mixed datasets that combined synthetic and real upcalls. Our findings demonstrated that while models trained exclusively on synthetic data were already as effective as those trained on real data for detecting and classifying real NARW upcalls, their performance greatly improved by having access to even small amounts of extra real vocalizations during training. These findings reinforce the importance of incorporating real vocalizations in the training process when possible. This approach enables a DNN trained on only synthetic vocalizations to find real vocalizations, thereby producing better datasets with each iteration.

7.1 Future Research

This Section outlines a number of potential research ideas and future work in underwater bioacoustic classification and detection.

7.1.1 Data Augmentation based on Deep Learning

The improved performance gained by increasing the diversity of the dataset through basic augmentation steps as done here opens up the possibility of applying more sophisticated data augmentation methods based on deep generative methods to generate marine mammal vocalizations. While our proposed solution is well-suited to the task of inflating the training dataset based on the distribution of already existing samples through either geometric transformations or linear superposition, it is limited by the type of transformations. Furthermore, SpecAugment itself is limited to only generating spectrograms and not the waveform itself. Deep generative algorithms however, are capable of creating new examples of the data that were not seen in the training dataset, but could have theoretically been drawn from the original dataset distribution. Such algorithms have been extensively explored in speech and image related tasks but their application to underwater bioacoustics is still incipient and are an interesting direction for future work.

Auto-Regressive Models. Generative models such as WaveNet [99] or SampleRNN [88] for audio generation are capable of modeling raw audio, and have been used to produce highly realistic speech examples conditioned on linguistic features. WaveNets

and SampleRNN are auto-regressive models that generate raw audio in a sample-by-sample manner, where each generated sample is conditioned on all previous ones. This presents two initial advantages, the technique produces more natural sounding output and provides full control on what kind of waveform is being generated. The main disadvantage of auto-regressive approaches for audio generation is their inherently slow inference time, which greatly limits their usability in real-time tasks such as text-to-speech synthesis. However, animal vocalization synthesis to increment a training dataset does not have this time constraint, which greatly expands their usability.

GANs-based approaches. Generative Adversarial Networks have emerged as a powerful and versatile tool with widespread adoption across various domains. In computer vision, GANs have been used for unconditional image generation [62] and video-to-video synthesis [19]. More recently, a number of works have begun to make progress in the use of GANs on the acoustic domain [29, 71], mainly towards text-to-speech synthesis. While audio quality has not significantly improved over the auto-regressive approaches, GANs based architectures have shown to be several time faster during audio generation, making them highly interesting for text-speech applications. In underwater bioacoustics, GANs have been successfully applied to generate odontocetes vocalizations [125] and dolphin whistles [76] as discussed in Chapter 2. Exploring the application of GANs for generating NARW upcalls presents an intriguing avenue for future research.

Neural Style Transfer. Neural Style Transfer techniques are able to manipulate images by applying the visual style of another image while preserving the original content. In other words these techniques are able to blend two images together where the resulting image looks and has the content of one image but is "painted" in the style of the other. Recently, similar "style" transfer techniques have begun to be explored in the context of acoustics, in particular to voice and musical instruments. Engel *et al.* [33] trained audio synthesis models built on top of the Differentiable Digital Signal Processing (DDSP) library [33] that are capable of timbre transfer between disparate sounds while keeping the frequency and loudness the same or even transfer of room acoustics to new environments.

In general, while few attempts have been made in generating animal vocalization with these types of methods, we believe that similar results to the augmentation techniques displayed in this work can be expected while opening up new possibilities. Under this perspective, a few paths for future research presents themselves. First, given the success of GANs for similar tasks, we plan to investigate the suitability of GANs for producing synthetic NARW upcalls. Second, inspired by computer vision's neural style transfer techniques and given the recent success presented by the DDPS library in acoustic transfer, we plan on investigating the augmentation potential of transferring marine mammal vocalizations to different underwater soundscapes and vice-versa. Other paths worth exploring are techniques adapted from video synthesis [140], and audio synthesis from few samples such as voice cloning [2] or few-shot image generation [77]. Video data presents similar characteristics to acoustics such as temporal features that could be adapted to sound generation while few-shot learning could help address the problem of acquiring large amounts of underwater acoustic data required to train effective generators.

The use of generative methods opens another path for using only synthetically generated vocalizations to train a DNN. It would be interesting to compare this approach to the methodology we introduced in phase 3 for generating synthetic vocalizations. However, while deep generative methods show great promise, they also offer limited control over what is generated. Therefore a validation stage of the generated data is recommend before being added to the training set. Unsupervised clustering techniques that could group embeddings of these generations into distinct groups for further analysis could keep manual validation efforts to a minimum. Designing a visual analytic interface where generated samples are grouped into high quality/low quality is an interesting possibility. Some preliminary investigation has been done in this direction where we employed a SampleRNN model to generate synthetic NARW upcalls. While the model was successfully generating realistic vocalizations, most of the synthetic samples contained only environmental noise leading to a highly skewed negative/positive generation ratio. Although the incorporation of a validation tool as described above could help select only high quality samples, we are looking at exploring different generative algorithms that would give a more consistent response.

7.1.2 Multi-Species detection and Classification

While the methodologies proposed in this study have proven effective for classifying NARW upcalls, they hold promise for the identification of vocalizations from other marine mammals as well. In this context, we plan to assess the methodology be developing a multi-species classifier for other low-frequency vocalizations. This step will also help evaluate the methodology's capability to discriminate between vocalizations that share similarities in duration, frequency band, and shape. One significant challenge when developing classifiers and detectors for NARW upcall detection is the extensive vocal repertoire of humpback whales. Certain sounds produced by humpbacks can closely resemble NARW upcalls, often leading to confusion, even among trained bioacousticians [26, 12]. In particular, incorporating humpback whale vocalizations into the NARW upcall detector will be a crucial step in demonstrating the capability of these DL-based detectors to accurately differentiate between the vocalizations of these two species.

7.1.3 Active Learning

While data augmentation and TL can drastically reduce the required amount of training data when developing a detection and classification system, these techniques cannot overcome all biases present in a small dataset. For instance, when adapting a model to a new task such as classifying blue whale calls, if there are no instances of blue whales in the original dataset, no amount of augmentation or TL will change this fact. The proposed approach described in Chapter 6 can be effective in bridging this gap and creating a first batch of detections to be analysed. In these cases, however, engaging with some annotation efforts by a human expert to validate the detections and extract the required segments is still necessary. One can further reduce the amount of time a human annotator would have to spend analysing the detections by employing active learning [89].

Active Learning [116] consists in having a DL model cleverly select the data that a human analyst is going to annotate, ensuring that the most useful samples for the task at hand are chosen. Through this strategy, the model can achieve a higher performance threshold using less data than randomly sampling. In this way, instead of having the human annotator validate all of the detection from the synthetic models,

we can ask the model itself to choose which of detections would be more valuable, thereby substantially reducing the amount training data required to achieve a satisfactory performance level. There are several active learning strategies that can be employed, with the most common ones being uncertainty sampling and query-by-committee (QBC). In uncertainty sampling [50], the model selects samples for which it is the most uncertain about its predictions. This can be measured in several ways, such as selecting samples with the least confidence as determined by the score outputted by the DNN for each class, or the smallest margin between the top two predicted class probabilities in a multi-class classification task. By annotating and incorporating these uncertain samples into the training set, the model learns to better handle ambiguous or borderline cases, ultimately improving its overall performance. QBC [124] is a multimodal appraoch where each model is a committee member. These are trained on the current annotated data. These models can be different instances of the same model, or completely different model architectures. The committee members then make predictions on the unlabeled samples, and samples with the highest disagreement among the committee members are selected for annotation. The idea behind QBC is similar to that of uncertainty sampling. We are addressing the uncertainty created between the disagreement between the predictions of a diverse set of models, each of which may capture different aspects of the data.

7.1.4 Incorporating Wider Temporal Context

The majority of studies that apply DL to underwater bioacoustics, including the present work, use spectrogram representations views of the audio data to train detectors and classifiers. However, these views are often temporally short (a few seconds long) and fail to consider information present in a wider temporal context as would be done by bioacousticians when analysing the data. While some approaches incorporate predictions in the immediate vicinity of a detection, such as Kirsebom *et al.* [65] where they average classification scores on overlapping sliding windows to detect NARW upcalls, it would be interesting to explore still more sophisticated DL approaches that consider an even wider temporal context.

In this context, Recurrent Neural Networks (RNNs) [146] have been employed in many tasks involving time-series due to their potential to capture sequential patterns

in larger temporal contexts. While a traditional RNN architecture could achieve similar results to what we have found in this work it would be interesting to explore sequential models [85, 42] that combines the abstraction powers of the convolutional layers of a CNN models with the sequential capabilities of recurrent layers. In this architecture the raw audio would serve as input to convolutional layers that would produce features for the recurrent layers, and could provide a better representation of the the data than a CNN or RNN architecture alone would.

While studies like Shiu *et al.* [127] and Foster [37] have investigated either the CNN or RNN approaches, and others like Ibrahim *et al.* [54] have provided a comparative analysis between the two, the exploration of architectures that combine both CNNs and RNNs remains incipient. Notably, there is a lack of consensus on which type architecture would be a better candidate for detecting marine mammals, or for sound detection in general. In fact, both Shiu *et al.* and Ibrahim *et al.* reported similar performance when using strictly a CNN or RNN architecture.

Sainath *et al.* [119] investigated the combination of CNNs and RNNs for voice search tasks. Their paper introduced the CLDNN architecture, merging CNN and Long Short-Term Memory (LSTM) components. This design utilizes CNN layers to transform the input signal into a variant set of features. Once transformed, the data progresses through LSTM layers for temporal modeling before being processed by fully connected layers to produce a refined feature representation. Similarly, within sound event detection, Çakır *et al.* [17] utilized a method that extracts features via multiple convolutional layers, equipped with filters spanning both time and frequency dimensions. These extracted features are subsequently fed as input into recurrent layers, and their outputs are used to determine event activity probabilities through a fully connected layer.

Building on the work of both Sainath *et al.* [119] and Çakır *et al.* [17], Madhusudhanan *et al.* [83] expanded upon the original concept within the realm of bioacoustics. Their architecture handles a series of fixed-size spectrograms totaling 2 minutes of audio, processed by pre-trained CNN layers. The resultant features are then used as input for LSTM layers. This combined network showed a 17% increase in F1-score performance for detecting fin whale calls compared to a standard CNN model. Notably, their work showed that the combined approach of CNN and RNN layers can

handle even larger temporal contexts. This is particularly promising for studying species known for their prolonged vocalizations or songs.

In terrestrial bioacoustics, Gupta *et al.* [47] proposed a similar approach for predicting and analyzing acoustics of 100 bird species using spectrograms. Their work uses CNNs for summarizing the spectral features with RNNs to capture temporal dependencies, yielding a model with an average accuracy of 67% across all bird species. The model employs a sliding window technique to process the spectrogram inputs through the CNNs, which are then concatenated before being fed into the RNN. The authors also showed various model combinations, including standalone CNNs, CNNs with LSTM networks, Gated Recurrent Units (GRU), and Legendre Memory Units (LMU).

A common theme across these studies is the combination of CNN's ability to extract specific features and RNN's capacity for modeling temporal sequences in one classifier. The experiments listed within each work show promise that these combined models generally perform better than standalone CNN or RNN models. This shift in approach, from specific "snapshot" detections to broader context analysis, provides deep neural networks with more temporal data to enhance their accuracy and sensitivity in bioacoustic detection tasks, ultimately leading to more robust and reliable analyses in complex acoustic environments.

Bibliography

[1] Roee Aharoni, Melvin Johnson, and Orhan Firat. Massively multilingual neural machine translation. *arXiv preprint arXiv:1903.00089*, 2019.

[2] Sercan O. Arik, Jitong Chen, Kainan Peng, Wei Ping, and Yanqi Zhou. Neural voice cloning with a few samples, 2018.

[3] Thierry Aubin, Pierre Jouventin, and Isabelle Charrier. Mother vocal recognition in antarctic fur seal arctocephalus gazella pups: a two-step process. *PLoS One*, 10(9):e0134513, 2015.

[4] Florian Aulanier, Yvan Simard, Nathalie Roy, Marion Bandet, and Cédric Gervaise. Groundtruthed probabilistic shipping noise modeling and mapping: Application to blue whale habitat in the gulf of st. lawrence. In *Proceedings of Meetings on Acoustics 4ENAL*, volume 27, page 070006. Acoustical Society of America, 2016.

[5] Mohammed Bahoura and Yvan Simard. Blue whale calls classification using short-time fourier and wavelet packet transforms and artificial neural network. *Digital Signal Processing*, 20(4):1256–1263, 2010.

[6] Mohammed Bahoura and Yvan Simard. Blue whale calls classification using short-time fourier and wavelet packet transforms and artificial neural network. *Digital Signal Processing*, 20(4):1256–1263, 2010.

[7] Max Bain, Arsha Nagrani, Daniel Schofield, Sophie Berdugo, Joana Bessa, Jake Owen, Kimberley J Hockings, Tetsuro Matsuzawa, Misato Hayashi, Dora Biro, et al. Automated audiovisual behavior recognition in wild primates. *Science advances*, 7(46):eabi4883, 2021.

[8] Mark F Baumgartner and Sarah E Mussoline. A generalized baleen whale call detection and classification system. *The Journal of the Acoustical Society of America*, 129(5):2889–2902, 2011.

[9] Christian Bergler, Manuel Schmitt, Rachael Xi Cheng, Andreas K Maier, Volker Barth, and Elmar Nöth. Deep learning for orca call type identification-a fully unsupervised approach. In *INTERSPEECH*, pages 3357–3361, 2019.

[10] Christian Bergler, Hendrik Schröter, Rachael Xi Cheng, Volker Barth, Michael Weber, Elmar Nöth, Heribert Hofer, and Andreas Maier. Orca-spot: An automatic killer whale sound detection toolkit using deep learning. *Scientific reports*, 9(1):1–17, 2019.

[11] Peter C Bermant, Michael M Bronstein, Robert J Wood, Shane Gero, and David F Gruber. Deep machine learning techniques for the detection and classification of sperm whale bioacoustics. *Scientific reports*, 9(1):1–10, 2019.

[12] Carolyn M Binder. *Impacts of environment-dependent acoustic propagation on passive acoustic monitoring of cetaceans*. PhD thesis, Dalhousie University, 2018.

[13] Carolyn M Binder and Paul Hines. Applying automatic aural classification to cetacean vocalizations. In *Proceedings of Meetings on Acoustics ECUA2012*, volume 17, page 070029. Acoustical Society of America, 2012.

[14] Carolyn M Binder and Paul C Hines. Range-dependent impacts of ocean acoustic propagation on automated classification of transmitted bowhead and humpback whale vocalizations. *The Journal of the Acoustical Society of America*, 145(4):2480–2497, 2019.

[15] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[16] Marco Brunoldi, Giorgio Bozzini, Alessandra Casale, Pietro Corvisiero, Daniele Grosso, Nicodemo Magnoli, Jessica Alessi, Carlo Nike Bianchi, Alberta Mandich, Carla Morri, et al. A permanent automated real-time passive acoustic monitoring system for bottlenose dolphin conservation in the mediterranean sea. *PloS one*, 11(1):e0145362, 2016.

[17] Emre Cakır, Giambattista Parascandolo, Toni Heittola, Heikki Huttunen, and Tuomas Virtanen. Convolutional recurrent neural networks for polyphonic sound event detection. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25(6):1291–1303, 2017.

[18] Thomas A Calupca, Kurt M Fristrup, and Christopher W Clark. A compact digital recording system for autonomous bioacoustic monitoring. *The Journal of the Acoustical Society of America*, 108(5):2582–2582, 2000.

[19] Caroline Chan, Shiry Ginosar, Tinghui Zhou, and Alexei A. Efros. Everybody dance now, 2019.

[20] Honglie Chen, Weidi Xie, Triantafyllos Afouras, Arsha Nagrani, Andrea Vedaldi, and Andrew Zisserman. Localizing visual sounds the hard way. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16867–16876, 2021.

[21] Jongkwon Choi, Youngmin Choo, and Keunhwa Lee. Acoustic classification of surface and underwater vessels in the ocean using supervised machine learning. *Sensors*, 19(16):3492, 2019.

[22] Min Jin Chong and David Forsyth. Gans n'roses: Stable, controllable, diverse image to image translation (works for videos too!). *arXiv preprint arXiv:2106.06561*, 2021.

[23] Enis Berk Çoban, Dara Pir, Richard So, and Michael I Mandel. Transfer learning from youtube soundtracks to tag arctic ecoacoustic recordings. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 726–730. IEEE, 2020.

[24] JG Cooke. Eubalaena glacialis. eubalaena glacialis. the iucn red list of threatened species 2020: e.t41712a162001243, 2020.

[25] Ayhan Dede, Ayaka Amaha Öztürk, Tomonari Akamatsu, Arda M Tonay, and Bayram Öztürk. Long-term passive acoustic monitoring revealed seasonal and diel patterns of cetacean presence in the istanbul strait. *Journal of the Marine Biological Association of the United Kingdom*, 94(6):1195–1202, 2014.

[26] J. Delarue, K.A. Kowarski, E.E. Maxner, J.T. MacDonnell, and S.B. Martin. Acoustic monitoring along canada's east coast: August 2015 to july 2017. *Document Number 01279, Environmental Studies Research Funds Report Number 215, Version 1.0. Technical report by JASCO Applied Sciences for Environmental Studies Research Fund, Dartmouth, NS, Canada*, 120, 2018.

[27] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[28] Simone Disabato, Giuseppe Canonaco, Paul G Flikkema, Manuel Roveri, and Cesare Alippi. Birdsong detection at the edge with deep learning. In *2021 IEEE International Conference on Smart Computing (SMARTCOMP)*, pages 9–16. IEEE, 2021.

[29] Chris Donahue, Julian McAuley, and Miller Puckette. Adversarial audio synthesis, 2019.

[30] Paul Nguyen Hong Duc. *Development of artificial intelligence methods for marine mammal detection and classification of underwater sounds in a weak supervision (but) Big Data-Expert context*. PhD thesis, Sorbonne Université, 12 2020.

[31] Emmanuel Dufourq, Carly Batist, Ruben Foquet, and Ian Durbach. Passive acoustic monitoring of animal populations with transfer learning. *Ecological Informatics*, 70:101688, 2022.

[32] Delphine Durette-Morin, Clair Evers, Hansen Johnson, Katie Kowarski, Julien Delarue, Hilary Moors-Murphy, Emily Maxner, Jack Lawson, and Kimberley Teresa Ann Davies. The distribution of north atlantic right whales in canadian waters from 2015-2017 revealed by passive acoustic monitoring. *Frontiers in Marine Science*, 9, 2022.

[33] Jesse Engel, Lamtharn Hantrakul, Chenjie Gu, and Adam Roberts. Ddsp: Differentiable digital signal processing, 2020.

[34] Christine Erbe, Rebecca Dunlop, and Sarah Dolman. Effects of noise on marine mammals. In *Effects of anthropogenic noise on animals*, pages 277–309. Springer, 2018.

[35] Mahdi Esfahanian, Nurgun Erdol, Edmund Gerstein, and Hanqi Zhuang. Two-stage detection of north atlantic right whale upcalls using local binary patterns and machine learning algorithms. *Applied Acoustics*, 120:158–166, 2017.

[36] Seppo Fagerlund. Bird species recognition using support vector machines. *EURASIP Journal on Advances in Signal Processing*, 2007(1):038637, 2007.

[37] Christopher M Foster. *Recurrent Neural Network Model for the Detection of Odontocetes*. PhD thesis, University of New Hampshire, 2022.

[38] Alexander N Gavrilov, Robert D McCauley, Chandra Salgado-Kent, Joy Tripovich, and Chris Burton. Vocal characteristics of pygmy blue whales and their change over time. *The Journal of the Acoustical Society of America*, 130(6):3651–3660, 2011.

[39] Jort F Gemmeke, Daniel PW Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R Channing Moore, Manoj Plakal, and Marvin Ritter. Audio set: An ontology and human-labeled dataset for audio events. In *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 776–780. IEEE, 2017.

[40] Cédric Gervaise, Yvan Simard, Florian Aulanier, and Nathalie Roy. *Optimal passive acoustic systems for real-time detection and localization of North Atlantic right whales in their feeding ground off Gaspé in the Gulf of St. Lawrence*. Department of Fisheries and Oceans, 2019.

[41] Burooj Ghani, Tom Denton, Stefan Kahl, and Holger Klinck. Feature embeddings from large-scale acoustic bird classifiers enable few-shot transfer learning. *arXiv preprint arXiv:2307.06292*, 2023.

[42] Soheila Gheisari, Sahar Shariflou, Jack Phu, Paul J Kennedy, Ashish Agar, Michael Kalloniatis, and S Mojtaba Golzan. A combined convolutional and recurrent neural network for enhanced glaucoma detection. *Scientific reports*, 11(1):1–11, 2021.

[43] Douglas Gillespie. Detection and classification of right whale calls using an edge detector operating on a smoothed spectrogram. *Canadian Acoustics*, 32:39–47, 2004.

[44] Douglas Gillespie. Dclde 2013 workshop dataset. *University of St Andrews Research Portal*, 2019.

[45] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[46] Anton Gradišek, Gašper Slapničar, Jure Šorn, Mitja Luštrek, Matjaž Gams, and Janez Grad. Predicting species identity of bumblebees through analysis of flight buzzing sounds. *Bioacoustics*, 26(1):63–76, 2017.

[47] Gaurav Gupta, Meghana Kshirsagar, Ming Zhong, Shahrzad Gholami, and Juan Lavista Ferres. Comparing recurrent convolutional neural networks for large scale bird species classification. *Scientific reports*, 11(1):17085, 2021.

[48] Danielle V. Harris, Jennifer L. Miksis-Olds, Julia A. Vernon, and Len Thomas. Fin whale density and distribution estimation using acoustic bearings derived from sparse arrays. *The Journal of the Acoustical Society of America*, 143(5):2980–2993, 2018.

[49] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[50] Tao He, Xiaoming Jin, Guiguang Ding, Lan Yi, and Chenggang Yan. Towards better uncertainty sampling: Active learning with multiple views for deep convolutional neural network. In *2019 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1360–1365. IEEE, 2019.

[51] Tyler A Helble, Gerald L D'Spain, John A Hildebrand, Gregory S Campbell, Richard L Campbell, and Kevin D Heaney. Site specific probability of passive acoustic detection of humpback whale calls from single fixed hydrophones. *The Journal of the Acoustical Society of America*, 134(3):2556–2570, 2013.

[52] Alex Hernández-García and Peter König. Further advantages of data augmentation on convolutional neural networks. In *International Conference on Artificial Neural Networks*, pages 95–103. Springer, 2018.

[53] Sicong Huang, Qiyang Li, Cem Anil, Xuchan Bao, Sageev Oore, and Roger B Grosse. Timbretron: A wavenet (cyclegan (cqt (audio))) pipeline for musical timbre transfer. *arXiv preprint arXiv:1811.09620*, 2018.

[54] Ali K Ibrahim, Hanqi Zhuang, Laurent M Chérubin, Michelle T Schärer-Umpierre, and Nurgun Erdol. Automatic classification of grouper species by their sounds using deep neural networks. *The Journal of the Acoustical Society of America*, 144(3):EL196–EL202, 2018.

[55] Ali K Ibrahim, Hanqi Zhuang, Nurgun Erdol, and Ali Muhamed Ali. A new approach for north atlantic right whale upcall detection. In *2016 International Symposium on Computer, Consumer and Control (IS3C)*, pages 260–263. IEEE, 2016.

[56] Hiroshi Inoue. Data augmentation by pairing samples for images classification. *arXiv preprint arXiv:1801.02929*, 2018.

[57] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015.

[58] Finn B Jensen, William A Kuperman, Michael B Porter, Henrik Schmidt, and Alexandra Tolstoy. *Computational Ocean Acoustics*, volume 2011. Springer, 2011.

[59] Jia-jia Jiang, Ling-ran Bu, Fa-jie Duan, Xian-quan Wang, Wei Liu, Zhong-bo Sun, and Chun-yue Li. Whistle detection and classification for whales based on convolutional neural networks. *Applied Acoustics*, 150:169–178, 2019.

[60] Hansen D Johnson, Christopher T Taggart, Arthur E Newhall, Ying-Tsong Lin, and Mark F Baumgartner. Acoustic detection range of right whale upcalls identified in near-real time from a moored buoy and a slocum glider. *The Journal of the Acoustical Society of America*, 151(4):2558–2575, 2022.

[61] Stefan Kahl, Connor M Wood, Maximilian Eibl, and Holger Klinck. Birdnet: A deep learning solution for avian diversity monitoring. *Ecological Informatics*, 61:101236, 2021.

[62] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks, 2019.

[63] Stephanie L King, Heidi E Harley, and Vincent M Janik. The role of signature whistle matching in bottlenose dolphins, tursiops truncatus. *Animal Behaviour*, 96:79–86, 2014.

[64] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014.

[65] Oliver S Kirsebom, Fabio Frazao, Yvan Simard, Nathalie Roy, Stan Matwin, and Samuel Giard. Performance of a deep neural network at detecting north atlantic right whale upcalls. *The Journal of the Acoustical Society of America*, 147(4):2636–2646, 2020.

[66] Amy R Knowlton and Scott D Kraus. Mortality and serious injury of northern right whales (eubalaena glacialis) in the western north atlantic ocean. *J. Cetacean Res. Manage.*, pages 193–208, 2001.

[67] Tom Ko, Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur. Audio augmentation for speech recognition. In *Sixteenth annual conference of the international speech communication association*, 2015.

[68] Daniel Kohlsdorf, Denise Herzing, and Thad Starner. An auto encoder for audio dolphin communication. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7. IEEE, 2020.

[69] Qiuqiang Kong, Yin Cao, Turab Iqbal, Yong Xu, Wenwu Wang, and Mark D Plumbley. Cross-task learning for audio tagging, sound event detection and spatial localization: Dcase 2019 baseline systems. *arXiv preprint arXiv:1904.03476*, 2019.

[70] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[71] Kundan Kumar, Rithesh Kumar, Thibault de Boissiere, Lucas Gestin, Wei Zhen Teoh, Jose Sotelo, Alexandre de Brebisson, Yoshua Bengio, and Aaron Courville. Melgan: Generative adversarial networks for conditional waveform synthesis, 2019.

[72] Mario Lasseck. Audio-based bird species identification with deep convolutional neural networks. *CLEF (working notes)*, 2125, 2018.

[73] Lei Li, Gang Qiao, Songzuo Liu, Xin Qing, Huaying Zhang, Suleman Mazhar, and Fuqiang Niu. Automated classification of tursiops aduncus whistles based on a depth-wise separable convolutional neural network and data augmentation. *The Journal of the Acoustical Society of America*, 150(5):3861–3873, 2021.

[74] Pu Li, Xiaobai Liu, Holger Klinck, Pina Gruden, and Marie A Roch. Using deep learning to track time× frequency whistle contours of toothed whales without human-annotated training data. *The Journal of the Acoustical Society of America*, 154(1):502–517, 2023.

[75] Pu Li, Xiaobai Liu, KJ Palmer, Erica Fleishman, Douglas Gillespie, Eva-Marie Nosal, Yu Shiu, Holger Klinck, Danielle Cholewiak, Tyler Helble, and Marie A Roch. Learning deep models from synthetic data for extracting dolphin whistle contours. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–10. IEEE, 2020.

[76] Pu Li, Marie A Roch, Holger Klinck, Erica Fleishman, Douglas Gillespie, Eva-Marie Nosal, Yu Shiu, and Xiaobai Liu. Learning stage-wise gans for whistle extraction in time-frequency spectrograms. *IEEE Transactions on Multimedia*, 2023.

[77] Yijun Li, Richard Zhang, Jingwan Lu, and Eli Shechtman. Few-shot image generation with elastic weight consolidation, 2020.

[78] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013.

[79] Jiantao Liu, Xiaoxiang Yang, Chen Wang, and Yi Tao. A convolution neural network for dolphin species identification using echolocation clicks signal. In *2018 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC)*, pages 1–4. IEEE, 2018.

[80] D.H. Loring and D.J.G. Nota. *Morphology and Sediments of the Gulf of St. Lawrence*. Bulletin (Fisheries Research Board of Canada). Fisheries and Marine Service, 1973.

[81] Tao Lu, Baokun Han, and Fanqianhui Yu. Detection and classification of marine mammal sounds using alexnet with transfer learning. *Ecological Informatics*, 62:101277, 2021.

[82] Wenyu Luo, Wuyi Yang, and Yu Zhang. Convolutional neural network for detecting odontocete echolocation clicks. *The Journal of the Acoustical Society of America*, 145(1):EL7–EL12, 2019.

[83] Shyam Madhusudhana, Yu Shiu, Holger Klinck, Erica Fleishman, Xiaobai Liu, Eva-Marie Nosal, Tyler Helble, Danielle Cholewiak, Douglas Gillespie, Ana Širović, et al. Improve automatic detection of animal call sequences with temporal context. *Journal of the Royal Society Interface*, 18(180):20210297, 2021.

[84] Yigit Mahmutoglu and Kadir Turk. A passive acoustic based system to locate leak hole in underwater natural gas pipelines. *Digital Signal Processing*, 76:59–65, 2018.

[85] Zelda Mariet and Vitaly Kuznetsov. Foundations of sequence-to-sequence modeling for time series. In *The 22nd international conference on artificial intelligence and statistics*, pages 408–417. PMLR, 2019.

[86] Tiago A. Marques, Len Thomas, Stephen W. Martin, David K. Mellinger, Jessica A. Ward, David J. Moretti, Danielle Harris, and Peter L. Tyack. Estimating animal population density using passive acoustics. *Biological Reviews*, 88(2):287–309, 2013.

[87] Trevor J McDougall and Paul M Barker. Getting started with teos-10 and the gibbs seawater (gsw) oceanographic toolbox. *Scor/Iapso WG*, 127(532):1–28, 2011.

[88] Soroush Mehri, Kundan Kumar, Ishaan Gulrajani, Rithesh Kumar, Shubham Jain, Jose Sotelo, Aaron Courville, and Yoshua Bengio. Samplernn: An unconditional end-to-end neural audio generation model, 2016.

[89] Kunal B Mehta, Jorge Rodriguez Saltijeral, Jesse Lopez, Abhishek Singh, Valentina Staneva, Scott Veirs, and Val Veirs. Active listening and learning for orca sound detection. *The Journal of the Acoustical Society of America*, 148(4):2728–2728, 2020.

[90] David K Mellinger. A comparison of methods for detecting right whale calls. *Canadian Acoustics*, 32:55–65, 2004.

[91] Zhong Meng, Yong Zhao, Jinyu Li, and Yifan Gong. Adversarial speaker verification. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6216–6220. IEEE, 2019.

[92] E Mercado III, SR Green, and JN Schneider. Understanding auditory distance estimation by humpback whales: a computational approach. *Behavioural processes*, 77(2):231–242, 2008.

[93] Nathan D Merchant, Matthew J Witt, Philippe Blondel, Brendan J Godley, and George H Smith. Assessing sound exposure from shipping in coastal waters using a single hydrophone and automatic identification system (ais) data. *Marine pollution bulletin*, 64(7):1320–1329, 2012.

[94] B Mishachandar and S Vairamuthu. Diverse ocean noise classification using deep learning. *Applied Acoustics*, 181:108141, 2021.

[95] Xavier Mouy, Mohammed Bahoura, and Yvan Simard. Automatic recognition of fin and blue whale calls for real-time monitoring in the st. lawrence. *The Journal of the Acoustical Society of America*, 126(6):2918–2928, 2009.

[96] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010.

[97] Juhan Nam, Keunwoo Choi, Jongpil Lee, Szu-Yu Chou, and Yi-Hsuan Yang. Deep learning for audio-based music classification and tagging: Teaching computers to distinguish rock from bach. *IEEE signal processing magazine*, 36(1):41–51, 2018.

[98] Inês Nolasco, Shubhr Singh, Veronica Morfi, Vincent Lostanlen, Ariana Strandburg-Peshkin, Ester Vidaña-Vila, Lisa Gill, Hanna Pamuła, Helen Whitehead, Ivan Kiskin, et al. Learning to detect an animal sound from five examples. *Ecological Informatics*, 77:102258, 2023.

[99] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[100] Julie N Oswald, Jay Barlow, and Thomas F Norris. Acoustic identification of nine delphinid species in the eastern tropical pacific ocean. *Marine mammal science*, 19(1):20–037, 2003.

[101] Hui Ou, Whitlow WL Au, and Julie N Oswald. A non-spectrogram-correlation method of automatically detecting minke whale boings. *The Journal of the Acoustical Society of America*, 132(4):EL317–EL322, 2012.

[102] Yassine Ouali, Céline Hudelot, and Myriam Tami. An overview of deep semi-supervised learning. *arXiv preprint arXiv:2006.05278*, 2020.

[103] Emma Ozanich, Aaron Thode, Peter Gerstoft, Lauren A Freeman, and Simon Freeman. Deep embedded clustering of coral reef bioacoustics. *The Journal of the Acoustical Society of America*, 149(4):2587–2601, 2021.

[104] Niall O'Mahony, Sean Campbell, Anderson Carvalho, Suman Harapanahalli, Gustavo Velasco Hernandez, Lenka Krpalkova, Daniel Riordan, and Joseph Walsh. Deep learning vs. traditional computer vision. In *Science and Information Conference*, pages 128–144. Springer, 2019.

[105] Federica Pace. *Comparison of feature sets for humpback whale song classification*. PhD thesis, Doctoral dissertation, MSc dissertation, University of Southampton, UK, 2008.

[106] Bruno Padovese, Fabio Frazao, Oliver S. Kirsebom, and Stan Matwin. Data augmentation for the classification of North Atlantic right whales upcallsa). *The Journal of the Acoustical Society of America*, 149(4):2520–2530, 04 2021.

[107] Bruno Padovese, Oliver S. Kirsebom, Fabio Frazao, Clair H.M. Evers, Wilfried A.M. Beslin, Jim Theriault, and Stan Matwin. Adapting deep learning models to new acoustic environments - a case study on the north atlantic right whale upcall. *Ecological Informatics*, page 102169, 2023.

[108] Daniel S Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D Cubuk, and Quoc V Le. Specaugment: A simple data augmentation method for automatic speech recognition. *arXiv preprint arXiv:1904.08779*, 2019.

[109] SE Parks, A Searby, A Célérier, MP Johnson, DP Nowacek, and PL Tyack. Sound production behavior of individual north atlantic right whales: implications for passive acoustic monitoring. *Endangered Species Research*, 15(1):63–76, 2011.

[110] Susan E Parks and Peter L Tyack. Sound production by north atlantic right whales (eubalaena glacialis) in surface active groups. *The Journal of the Acoustical Society of America*, 117(5):3297–3306, 2005.

[111] Luis Perez and Jason Wang. The effectiveness of data augmentation in image classification using deep learning. *arXiv preprint arXiv:1712.04621*, 2017.

[112] H.M. Pettis, R.M. III Pace, and P.K. Hamilton. North atlantic right whale consortium: 2022 annual report card. *Report to the North Atlantic Right Whale Consortium*, 2023.

[113] Mohammad Pourhomayoun, Peter Dugan, Marian Popescu, Denise Risch, Hal Lewis, and Christopher Clark. Classification for big dataset of bioacoustic signals based on human scoring system and artificial neural network. *arXiv preprint arXiv:1305.3633*, 2013.

[114] Michael S Reichert and Bernhard Ronacher. Noise affects the shape of female preference functions for acoustic signals. *Evolution*, 69(2):381–394, 2015.

[115] Jimmy Ren, Yongtao Hu, Yu-Wing Tai, Chuan Wang, Li Xu, Wenxiu Sun, and Qiong Yan. Look, listen and learn—a multimodal lstm for speaker identification. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[116] Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Brij B Gupta, Xiaojiang Chen, and Xin Wang. A survey of deep active learning. *ACM computing surveys (CSUR)*, 54(9):1–40, 2021.

[117] Marie Roch, Melissa Soldevilla, and John Hildebrand. Automatic species identification of odontocete calls in the southern california bight. *The Journal of the Acoustical Society of America*, 116(4):2614–2614, 2004.

[118] Rosalind M Rolland, Susan E Parks, Kathleen E Hunt, Manuel Castellote, Peter J Corkeron, Douglas P Nowacek, Samuel K Wasser, and Scott D Kraus. Evidence that ship noise increases stress in right whales. *Proceedings of the Royal Society B: Biological Sciences*, 279(1737):2363–2368, 2012.

[119] Tara N Sainath, Oriol Vinyals, Andrew Senior, and Haşim Sak. Convolutional, long short-term memory, fully connected deep neural networks. In *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 4580–4584. Ieee, 2015.

[120] Justin Salamon and Juan Pablo Bello. Deep convolutional neural networks and data augmentation for environmental sound classification. *IEEE Signal processing letters*, 24(3):279–283, 2017.

[121] Laela Sayigh, Mary Ann Daher, Julie Allen, Helen Gordon, Katherine Joyce, Claire Stuhlmann, and Peter Tyack. The watkins marine mammal sound database: an online, freely accessible resource. In *Proceedings of Meetings on Acoustics*, volume 27. AIP Publishing, 2016.

[122] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

[123] OM Serra, FPR Martins, and Linilson Rodrigues Padovese. Active contour-based detection of estuarine dolphin whistles in spectrogram images. *Ecological Informatics*, 55:101036, 2020.

[124] H Sebastian Seung, Manfred Opper, and Haim Sompolinsky. Query by committee. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 287–294, 1992.

[125] Saumil Mehulbhai Shah. *Learning to Detect Odontocete Whistles from Generative Samples*. PhD thesis, San Diego State University, 2020.

[126] Sarah M Sharp, William A McLellan, David S Rotstein, Alexander M Costidis, Susan G Barco, Kimberly Durham, Thomas D Pitchford, Katharine A Jackson, P-Y Daoust, Tonya Wimmer, et al. Gross and histopathologic diagnoses from north atlantic right whale eubalaena glacialis mortalities between 2003 and 2018. *Diseases of Aquatic Organisms*, 135(1):1–31, 2019.

[127] Yu Shiu, KJ Palmer, Marie A Roch, Erica Fleishman, Xiaobai Liu, Eva-Marie Nosal, Tyler Helble, Danielle Cholewiak, Douglas Gillespie, and Holger Klinck. Deep neural networks for automated detection of marine mammal species. *Scientific Reports*, 10(1):1–12, 2020.

[128] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):60, 2019.

[129] Yvan Simard and Nathalie Roy. Detection and localization of blue and fin whales from large-aperture autonomous hydrophone arrays: A case study from the st. lawrence estuary. *Canadian Acoustics*, 36(1):104–110, 2008.

[130] Yvan Simard, Nathalie Roy, Samuel Giard, and Florian Aulanier. North atlantic right whale shift to the gulf of st. lawrence in 2015, revealed by long-term passive acoustics. *Endangered Species Research*, 40:271–284, 2019.

[131] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[132] Joy E Stanistreet, Douglas P Nowacek, Simone Baumann-Pickering, Joel T Bell, Danielle M Cholewiak, John A Hildebrand, Lynne EW Hodge, Hilary B Moors-Murphy, Sofie M Van Parijs, and Andrew J Read. Using passive acoustic monitoring to document the distribution of beaked whale species in the western north atlantic ocean. *Canadian Journal of Fisheries and Aquatic Sciences*, 74(12):2098–2109, 2017.

[133] Larissa Sayuri Moreira Sugai, Thiago Sanna Freire Silva, Jr Ribeiro, José Wagner, and Diego Llusia. Terrestrial Passive Acoustic Monitoring: Review and Perspectives. *BioScience*, 69(1):15–25, 11 2018.

[134] Jessie C Tanner, Joshua Justison, and Mark A Bee. Synsing: open-source matlab code for generating synthetic signals in studies of animal acoustic communication. *Bioacoustics*, 29(6):731–752, 2020.

[135] Luke Taylor and Geoff Nitschke. Improving deep learning with generic data augmentation. In *2018 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1542–1547. IEEE, 2018.

[136] Mark Thomas, Bruce Martin, Katie Kowarski, Briand Gaudet, and Stan Matwin. Marine mammal species classification using convolutional neural networks and a novel acoustic representation. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 290–305. Springer, 2019.

[137] Leslie G Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

[138] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[139] William Vickers, Ben Milner, Denise Risch, and Robert Lee. Robust north atlantic right whale detection using deep learning models for denoising. *The Journal of the Acoustical Society of America*, 149(6):3797–3812, 2021.

[140] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Generating videos with scene dynamics, 2016.

[141] Emily E Waddell, Jeppe H Rasmussen, and Ana Širović. Applying artificial intelligence methods to detect and classify fish calls from the northern gulf of mexico. *Journal of Marine Science and Engineering*, 9(10):1128, 2021.

[142] Janice M Waite, Kate Wynne, and David K Mellinger. Documented sighting of a north pacific right whale in the gulf of alaska and post-sighting acoustic monitoring. *Northwestern Naturalist*, pages 38–43, 2003.

[143] Zuxuan Wu, Xi Wang, Yu-Gang Jiang, Hao Ye, and Xiangyang Xue. Modeling spatial-temporal clues in a hybrid deep learning framework for video classification. In *Proceedings of the 23rd ACM international conference on Multimedia*, pages 461–470, 2015.

[144] Kele Xu, Dawei Feng, Haibo Mi, Boqing Zhu, Dezhi Wang, Lilun Zhang, Hengxing Cai, and Shuwen Liu. Mixup-based acoustic scene classification using multi-channel convolutional neural network. In *Pacific Rim Conference on Multimedia*, pages 14–23. Springer, 2018.

[145] Wuyi Yang, Wenyu Luo, and Yu Zhang. Classification of odontocete echolocation clicks using convolutional neural network. *The Journal of the Acoustical Society of America*, 147(1):49–55, 2020.

[146] Yong Yu, Xiaosheng Si, Changhua Hu, and Jianxun Zhang. A review of recurrent neural networks: Lstm cells and network architectures. *Neural computation*, 31(7):1235–1270, 2019.

[147] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

[148] Zhong-Qiu Zhao, Peng Zheng, Shou-tao Xu, and Xindong Wu. Object detection with deep learning: A review. *IEEE transactions on neural networks and learning systems*, 30(11):3212–3232, 2019.

[149] Ming Zhong, Manuel Castellote, Rahul Dodhia, Juan Lavista Ferres, Mandy Keogh, and Arial Brewer. Beluga whale acoustic signal classification using deep learning neural network models. *The Journal of the Acoustical Society of America*, 147(3):1834–1841, 2020.

[150] Ming Zhong, Jack LeBien, Marconi Campos-Cerqueira, Rahul Dodhia, Juan Lavista Ferres, Julian P Velev, and T Mitchell Aide. Multispecies bioacoustic classification using transfer learning of deep convolutional neural networks with pseudo-labeling. *Applied Acoustics*, 166:107375, 2020.

Appendix A

Acoustic Propagation Modelling

Technical aspects of how sound propagation is modelled in the Kadlu package is presented here. The following was extracted from the package's documentation and is based on the "Computational Ocean Acoustics" book by Jensen *et al.* [58].

Kadlu Sound Propagation

Oliver S. Kirsebom

March 2020

1 The PE Equation

Kadlu uses a Parabolic Equation (PE) approch for solving the wave equation (Jensen, Ch. 6). This yields solutions that are valid in the farfield (distance from source considerable greater than wavelength) for environments that exhibit "weak" range dependence. More specifically, Kadlu implements a numerical solution to the PE equation introduced by Thomson and Chapman,

$$\frac{\partial \psi}{\partial r} = ik_0 \left(n - 2 + \left[1 + k_0^{-2} \frac{\partial^2}{\partial z^2} \right]^{1/2} \right) \psi \tag{1}$$

which is said to have "good wide-angled behavior for realistic ocean acoustic environments with moderate changes in the refraction index", $n \equiv c_0/c$. Rearranging and introducing $A \equiv ik_0(n-1)$ and $B \equiv ik_0[-1+(1+k_0^{-2}\frac{\partial^2}{\partial z^2})^{1/2}]$, this can be written in the compact form,

$$\frac{\partial \psi}{\partial r} = (A + B)\psi \tag{2}$$

2 Split-Step Fourier Solution

Following Jensen Eq. (6.123), we approximate the solution with,

$$\psi(r + \Delta r) \;\approx\; e^{\frac{B}{2}\Delta r} e^{A\Delta r} e^{\frac{B}{2}\Delta r} \psi(r) \tag{3}$$

Finally, using $\mathcal{F}$ to denote the fourier transform $z \rightarrow k_z$ and using the correspondence $\frac{\partial^2}{\partial z^2} \rightarrow -k_z^2$ (Jensen Eq. (6.87)), we obtain the split-step Fourier formula,

$$\mathcal{F}\psi(r + \Delta r, z) \;\approx\; U_{\mathrm{D}}(\tfrac{1}{2}\Delta r)\, \mathcal{F}\, U_{\mathrm{R}}(\Delta r)\, \mathcal{F}^{-1}\, U_{\mathrm{D}}(\tfrac{1}{2}\Delta r)\, \mathcal{F}\psi(r, z)\;, \tag{4}$$

where U_{D} and U_{R} are the *diffractive* and *refractive* propagation matrices, respectively,

$$\begin{aligned} U_{\mathrm{D}}(x) \;&=\; e^{i\left[\left(k_0^2 + k_z^2\right)^{1/2} - k_0\right]x}\;, \\ U_{\mathrm{R}}(x) \;&=\; e^{ik_0(n-1)x} \end{aligned} \tag{5}$$

3 Computational domain

Following Jensen Sec. 6.5.3, we implement the split-step Fourier algorithm on a uniform grid $(\Delta r, \Delta z)$. Following Jensen, we adopt a default grid size of,

$$\Delta z = \lambda/2 \ , \quad \Delta r = 2\Delta z \ , \quad \lambda \equiv c_0/f = 2\pi/k_0 \ , \tag{6}$$

where $c_0 = 1,500$ m/s is the reference sound speed in water. (The option is provided for the user to specify a finer/coarser grid as needed.) The water surface ($z = 0$) is treated as a pressure-release surface, requiring $\psi(r, 0) = 0$. At the bottom, we terminate the physical solution domain by an artificial absorption layer of uniform thickness and a complex index of refraction of the form,

$$n^2 = n_b^2 + i\alpha e^{-(|z|-z_{\max})^2/D^2} \ , \tag{7}$$

where we adopt $\alpha = 1/(\pi \log_{10} e) \approx 0.733$, $D = (z_{\max} - H)/3$, and $z_{\max} = \frac{4}{3}H$. We determine the depth at which the physical domain is terminated, H, from the requirement that the real bottom should have a thickness of at least several wavelengths. Thus, we take,

$$H = \max z_b + 3\lambda \ , \tag{8}$$

where $\max z_b$ is the maximum seafloor depth in the domain.

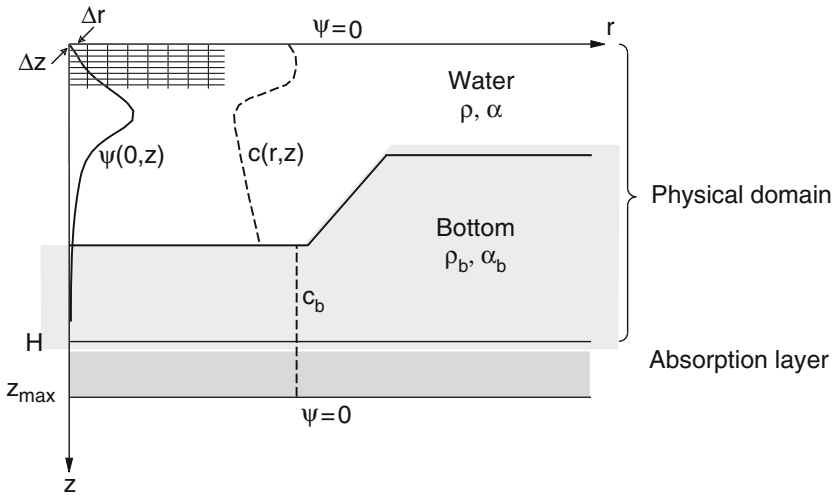

Figure 1: Shematic of PE solution domain (adapted from Jensen).

4 Water-Bottom Interface

For realistic treatment of bottom effects on sound propagation, it is important to include density changes at the water-bottom interface. We follow the approach described in Jensen Sec. 6.5.4. The refractive index, n, is replaced with the *effective* index of refraction,

$$\widetilde{n}^2 = n^2 + \frac{1}{2k_0^2}\left[\frac{1}{\rho}\nabla^2\rho - \frac{3}{2\rho^2}(\nabla\rho)^2\right] , \tag{9}$$

and the displacement potential ψ in Eqs. (1)–(4) is replaced with $\widetilde{\psi} = \psi/\sqrt{\rho}$. We remove density discontinuities at the water-bottom interface by introducing a smoothing function of the form,

$$\rho(z) = \rho + \tfrac{1}{2}(\rho_b - \rho)\tanh\chi , \tag{10}$$

where $\chi \equiv \frac{|z|-z_b}{L}$, z_b being the seafloor depth and L the distance over which the density changes from ρ to ρ_b. We adopt $L = \pi/k_0 = \lambda/2$, close to the value of $L = 2/k_0$ suggested by Jensen. By taking the gradient of Eq. (10), we further obtain,

$$|\nabla\rho| \;=\; \frac{1}{2L}(\rho_b - \rho)\,\mathrm{sech}^2\chi\left[1 + (\nabla z_b)^2\right]^{1/2} , \tag{11}$$

$$\nabla^2\rho \;\approx -\frac{1}{L^2}(\rho_b - \rho)\,\mathrm{sech}^2\chi\tanh\chi\left[1 + (\nabla z_b)^2\right] \tag{12}$$

where $(\nabla z_b)^2 = \left(\frac{\partial z_b}{\partial r}\right)^2 = \cos^2\phi\left(\frac{\partial z_b}{\partial x}\right)^2 + \sin^2\phi\left(\frac{\partial z_b}{\partial y}\right)^2$. Moreover, in the second equation, we have neglected the curvature of the seafloor, i.e., $\nabla^2 z_b \approx 0$. We note that Kadlu assumes a single bottom layer, although it would be straightforward to generalize the implementation to handle several layers.

5 Volume Attenuation

We ignore volume attenuation in the water column, but include volume attenuation in the bottom layer by subtracting a complex term from the sound speed, c_b. The complex term is computed as the largest, real root of the polynomial $\beta x^2 - x + \beta c_b^2 = 0$ fulfilling $0 \le x < c_b$. Here $\beta = \alpha_b^{(\lambda)}/(40\pi c_b \log_{10} e)$ with $\alpha_b^{(\lambda)}$ being the attenuation coefficient in units of dB/λ. For typical values of $\alpha_b^{(\lambda)}$, this leads to,

$$c \;\to\; c_b(1 - i\beta c_b) \tag{13}$$

$$n_b^2 \;\to\; n_b^2(1 + i2\beta c_b) \tag{14}$$

6 Starter

We use the Thomson starter field, as defined in Jensen Sec. 6.4.2.3,

$$\psi(0,k_z) \;=\; \begin{cases} \frac{e^{-i\pi/4}}{\Delta z}\,(8\pi)^{1/2}\left(k_0^2-k_z^2\right)^{-1/4}\sin k_z z_s\;, & |k_z|<k_0\sin\theta_1 \\ 0\;, & |k_z|\geq k_0\sin\theta_1 \end{cases} \tag{15}$$

where $\psi(0,k_z)\equiv\mathcal{F}\psi(0,z)$, while z_s and θ_1 are the depth and half-beamwidth of the source, respectively, and Δz is the vertical grid spacing. Note that the prefactor $\frac{e^{-i\pi/4}}{\Delta z}$ does not appear in the formula given in Jensen Sec. 6.4.2.3.