Martin Moan

Noise in the Sea

Deep Learning for Marine Acoustic Data

Master's thesis in Computer Science
Supervisor: Özlem Özgöbek
Co-supervisor: Luca Tassara
July 2022

Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

Martin Moan

Noise in the Sea

Deep Learning for Marine Acoustic Data

Master's thesis in Computer Science
Supervisor: Özlem Özgöbek
Co-supervisor: Luca Tassara
July 2022

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

Abstract

Increased human activity in the oceans has transformed the pre-industrial marine acoustic environments through significant anthropogenic sound pollution from activities such as shipping, seismic survey equipment and pile driving, to name a few. Many marine animals are acoustic specialists, using sound as their main method of perception of their environment and communication with conspesific individuals. Anthropogenic noise has been found to affect marine wildlife in a number of ways, including auditory masking, increased stress, behavioural changes and physiological damage. Therefore it is vital that we monitor the acoustic environment of our oceans, to ensure conservation efforts are effective in preserving the health of our marine ecosystems. For this Unmanned glider-based Autonomous Underwater Vehicles (AUVs) provide a promising platform for acoustic monitoring due to their relatively low cost, and their ability to be deployed for long periods of time while traversing the marine environment, while introducing minimal noise into the environment compared to mobile ship-based recording platforms. In this thesis three deep learning based models are applied and evaluated for the task of multi-label classification of glider-based acoustic data, to detect and classify anthropogenic and biological sounds in marine environments. We train the ResNet18 convolutional architecture, and the Audio Spectrogram Transformer model using supervised techniques. The Audio Spectrogram Transformer is also pretrained using a self-supervised framework, and fine-tuned for the same multi-label classification task. The supervised and self-supervised models are compared to investigate the effect of both increased model complexity and self-supervised pretraining on the performance of the final multi-label classification task. The models are evaluated using several common metrics in the audio classification domain, including F1 Score, mean Average Precision (mAp) and Area Under the Receiver Operating Characteristic curve (AUROC). The results presented in this thesis show that the supervised Audio Spectrogram Transformer is the most capable in the classification task among the three models used in this project, achieving mAp and AUROC of approximately 95.7 and 99.1 respectively. The results also show that despite its reduced complexity compared to the Audio Spectrogram Transformer, the ResNet18 model achieves good results with mAp and AUROC of approximately 92.8 and 99.2 respectively. The self-supervised Audio Spectrogram Transformer however performs significantly worse on the final multi-label classification task, achieving mAp and AUROC of approximately 43.2 and 82.5 respectively. Indicating that the supervised frameworks can achieve better results for the final prediction task than the self-supervised pretrained models, if enough labeled data is available for training. The results of the self-supervised model indicate that models trained in this framework can indeed be used for sound event detection and classification of marine acoustic data if labeled data is significantly limited or missing, although being outperformed by the purely supervised models. The results of this thesis also show that transformer based audio classification models, which have achieved promising results in the terrestrial domain, are also capable of audio classification of glider-based marine acoustic data.

Sammendrag

Økt menneskelig aktivitet har forandret havets akustiske miljø fra før-industriell tid gjennom vesentlig antropogenisk lydforurensning, fra aktiviteter som for eksempel shipping, bruk av geologisk undersøkelsesutstyr og påling. Mange marine arter er spesialister på lyd, og benytter lyd som sitt primære verktøy for å oppfatte sin omverden og for å kommunisere med andre individer av samme art. Det er blitt funnet at antropogenisk støy påvirker marine dyr på diverse måter, blant annet gjennom auditiv maskering, økt stress, adferdsmessige forandringer samt fysiologiske skader. Overvåkning av marine akustiske miljø er derfor avgjørende for å tilse effektivt vern av marine økosystem. Ubemannede glider baserte autonome undervannsfarkoster (AUV) kan være lovende plattformer for å gjennomføre denne typen overvåkning, grunnet relativt lave kostnader forbundet med disse plattformene i forhold til for eksempel skipsbårent akustisk utstyr. Samt også grunnet i at disse plattformene kan utplasseres over lengre tidsrom og traversere havområder, uten å produsere vesentlig støy i forhold til skipsbårent akustisk utstyr. I denne avhandlingen benytter og evaluerer vi tre modeller som benytter dyplæringsteknikker for å klassifisere antropogeniske og biologiske lyder i marine lyddata fra glider baserte AUV plattformer. I dette arbeidet benytter vi ResNet18 CNN arkitekturen og Audio Spectrogram Transformer (AST) transformator nettverket, trent gjennom veiledet læring. Vi trener også AST modellen gjennom selv-veiledet forhåndstrening (SSAST) som deretter fintrenes for den samme klassifiseringsoppgaven. Både de veiledede og selv-veiledede modellene sammenlignes for å undersøke effekten av både økt kompleksitet i modellene og selv-veiledning har på modellenes ytelsesevne i klassifiseringsoppgaven. Modellene evalueres ved hjelp av flere mål som ofte benyttes innen lydklassifisering, blant annet F1 verdi, "mean Average Precision (mAp)" og "Area Under the Receiver Operating Characteristic curve (AUROC)". Resultatene som presenteres i denne oppgaven viser at AST modellen som oppnår beste ytelse i klassifiseringsoppgaven, og oppnår omtrentlig mAp og AUROC på henholdsvis 95.7 og 99.1. Resultatene viser videre at ResNet18 modellen, til tross for vesentlig lavere kompleksitet i forhold til AST, oppnår overraskende gode resultater, med omtrentlig mAp og AUROC på henholdsvis 92.8 og 99.2. SSAST yter vesentlig verre derimot, og oppnår kun omtrentlige mAp of AUROC på henholdsvis 43.2 og 82.5. Som kan tyde på at de veiledede modellene egner seg bedre til den endelige klassifiseringsoppgaven, dersom tilstrekkelige markerte data er tilgjengelig for trening. Resultatene av den selv-veiledede modellen kan tyde på at slike modeller *kan* benyttes for lyddeteksjon og klassifisering av marine akustiske data dersom tilstrekkelig mengde markerte treningsdata ikke er tilgjengelig, men at disse kan bli vesentlig forbigått av veiledede modeller. Resultatene presentert i denne oppgaven viser også tydelig at dyplæringsmodeller basert på transformator arkitekturen, som har oppnådd lovende resultater for landbasert lydklassifisering, også egner seg godt for klassifisering av marine lyddata fra glider baserte AUV plattformer.

Preface

This thesis is submitted in partial fulfillment of the degree of Master of Science in Engineering at the Department of Computer Science (IDI) at Norwegian University of Science and Technology (NTNU), as part of the course *TDT4900 - Computer Science, Master's Thesis*. The work was guided by the project supervisor Özlem Özgöbek, associate Professor at IDI at NTNU, and was conducted in cooperation with Akvaplan Niva, under the supervision by Luca Tassara. The work presented in this thesis is a continuation of our previous work from [1].

Acknowledgements

Firstly, I would like to thank my supervisor Özlem Özgöbek at NTNU for guidance, valuable feedback, for challenging me in my assumptions and for making me reflect on my work. I would also like to thank the external project supervisor Luca Tassara for the opportunity and freedom to explore deep learning in the marine audio domain, and for guidance and feedback on my research. I am also grateful for the advice and guidance regarding marine acoustics given to me by Ana Širović, associate professor at the Department of Biology at NTNU. In addition I would like to thank the people of the High Performance Computing Group at NTNU for providing me with the computing resources of the Idun computing cluster and their guidance, that enabled me to apply state-of-the-art deep learning models in my work in marine acoustics. And Håkon Hukkelås, PhD Candidate at IDI at NTNU for advice on how to utilize these resources efficiently. I would also like to thank my family for proofreading my thesis, and for all their support through several years of study, particularly this past year.

Finally I thank my fiancé, Linnea Hoff Johannessen for all your unwavering support, immense patience, excellent advise and boundless kindness. Without you I would not have been able to complete this project, and I would not be the person that I am today.

Contents

Figures

Tables

Glossary

Akvalplan-NIVA A daughter company of the Norwegian Institute for Water Research (nor.: Norsk Institutt for Vannforskning). 3

Aliasing An artefact of sampled signals in which the signal contains frequencies that are too high for the sampling rate to sufficiently reconstruct.. 12

Anthropogenic Something originating from human activity [2]. 2, 5

Bioacoustic The study of how biological organisms produce and receive sound [3]. 3

Biophonic Sounds originating from animals [4]. 5

Deep Learning A subset of Machine Learning based models, specifically neural networks that span multiple layers.. 4

Echolocation A form of sonar in which the reflected echoes of a produced sound are used to localise and identify objects in the environment [5]. 32

Mel scale A scale that tries to map changes in physical frequencies to how these changes are perceived by humans [6]. 17

Ocean Networks Canada An initiative from the University of Victoria (Canada), operating several ocean observatories [7]. 30

Octave An interval between two frequencies where one frequency is double that of the other [8]. 17

Pitch Pitch describes how humans *perceive* frequencies of sound [9]. 17

SLURM Slurm Workload Manager [10]. 53, 61

Soniferous Something that produces sound [11]. 2

Spectrogram A time-frequency representation of some signal. 14

Transformer In the field of machine learning, the transformer is a type of nerual network that solely relies on the attention mechanism. 1, 4

Unsupervised In the context of machine learning, this refers to the how a model can be trained using data without any known desired output values.. 5

Acronyms

AST	Audio Spectrogram Transformer. v, xv, 1, 4, 5, 6, 7, 20, 26, 34, 35, 37, 43, 64
AUROC	Area Under the Receiver Operating Characteristic curve. iii, v, 30, 31, 32, 52, 64
AUV	Autonomous Underwater Vehicle. iii, v, vi, 1, 2, 4, 20, 56, 67
CD	Compact Disk. 11
CNN	Convolutional Neural Network. xv, 1, 4, 5, 6, 29, 30, 31, 64
CPU	Central Processing Unit. 53
CRNN	Convolutional Recurrent Neural Network. 24, 29
dB	decibel. 17
DBSCAN	Density-Based Spatial Clustering of Applications with Noise. 31
DCASE	Detection and Classification of Acoustic Scenes and Events. 36
DCLDE	Detection Classification Localization and Density Estimation. xv, 6, 30, 36
DFT	Discrete Fourier Transform. xiii, 15
DSC	Deep Sound Channel. 19
FPR	False Positive Rate. 56
GMM	Gaussian Mixture Model. 31, 34
GPU	Graphics Processing Unit. 53
HDBSCAN	Hierarchical Density-Based Spatial Clustering of Applications with Noise. 31
HMM	Hidden Markov Model. 34

HPC High Performance Computing. 53

I/O Input/Output. 43

IDI Department of Computer Science. vii, ix

mAp mean Average Precision. iii, v, 34, 35, 52, 64

MFCC Mel Frequency Cepstral Coefficients. 31, 34

ML Machine Learning. 4, 5

MLOps Machine Learning Operations. 54

MLP Multilayer Perceptron. 28

MSE Mean Squared Error. 28

MSPM Masked Spectrogram Patch Modeling. 49

NEMO-ONDE NEutrino Mediterranean Observatory – Ocean Noise Detection Experiment. 32

NLP Natural Language Processing. 4, 23, 26

NOAA National Oceanic and Atmospheric Administration. 36

NTNU Norwegian University of Science and Technology. vii, ix, 53

PAM Passive Acoustic Monitoring. 1, 2

PCA Principal Component Analysis. 30

PIFSC Pacific Islands Fisheries Science Center. 36

PL PyTorch Lightning. 52, 53

QDFA Quadratic Discriminant Function Analysis. 33

RGB Red Green Blue. 27

RNN Recurrent Neural Network. 24, 29, 30

SED Sound Event Detection. 6

SOFAR Sound Fixing and Ranging channel. 19

SOTA state-of-the-art. 1, 4, 5, 26, 35

SSAST Self-Supervised Audio Spectrogram Transformer. xv, 1, 5, 6, 7, 20, 34, 35, 37, 64

STFT Short-Time Fourier Transform. 12, 15, 16, 65

SVM Support Vector Machine. 30

TPR True Positive Rate. 56

ViT Vision Transformer. 4, 26, 34, 43

WandB Weights and Biases. 54

ZCR Zero-Crossing Rate. 31

Chapter 1

Introduction

Acoustic monitoring of marine environments can help researchers in providing policy-makers with necessary knowledge to ensure effective conservation efforts of our oceans and their wildlife. As the effects of human activities continue to expand within these environments, effective conservation efforts have become vital in ensuring the health of our marine ecosystems. Passive Acoustic Monitoring (PAM) are techniques that enable monitoring of both marine and terrestrial environments through analysis of passively recorded sound. By using autonomous mobile recording platforms, such as the glider based Autonomous Underwater Vehicle (AUV), the spatial range at which acoustic monitoring can be performed can be significantly increased compared to static recording platforms. Furthermore, gliders avoid introducing significant noise into the environment that can disturb local wildlife and pollute the collected audio data, which can occur from engine noise with ship-towed hydrophone arrays.

For researchers in the relevant fields to gleam insights into these environments using PAM, the recorded audio must be processed to detect and classify the relevant sounds found in the recordings, before further analysis. The Convolutional Neural Network (CNN) architecture has been common in this task for both marine and terrestrial acoustic monitoring[12]. However, since its inception in 2017 the Transformer neural network architecture have provided state-of-the-art (SOTA) performance in a variety of tasks, including audio classification of terrestrial audio data [14, 15]. To the best of our knowledge, no transformer-based models have been evaluated for marine acoustic data collected using gliders.

In this thesis the Audio Spectrogram Transformer (AST)[14], a purely transformer-based deep learning model for audio data, is applied to the task of classification of marine acoustic data collected using glider AUVs. The model is trained using both supervised and self-supervised frameworks, as performed in the original AST and Self-Supervised Audio Spectrogram Transformer (SSAST) papers. The resulting models are then compared to the performance of CNN-based model used in the author's preliminary work in [1].

The github repository for this project can be found here: `https://github.com/MartinMoan/TDT4900-Noise-in-the-Sea-Source/tree/main`

1.1 Background and Motivation

In the ocean, sound can travel far more effectively than in terrestrial environments, making sound a far more effective tool for long range communication than typical electromagnetic mediums such as radio in marine environments. The effectiveness of acoustic propagation in marine environments is decided by a number of factors, such as salinity and temperature of the traversed body of water. These are affected by depth, latitude and longitude.

As human activities continue to expand, their effects on natural wildlife increase in turn. One such effect is that of sound pollution from Anthropogenic sources. For example propeller noise of shipping, pile driving or explosions of acoustic geological survey devices called *airguns*. Such anthropogenic noise has been found to affect the ability of some marine animals to discern biologically important acoustic signals, cause auditory masking, increased stress, behavioural changes and physiological damage [16–18]. As the low frequency sounds produced by these activities can travel long distances in the marine environment, large areas can be affected by it. In turn, this exacerbates the negative effects anthropogenic sound pollution in the ocean has on wildlife, as the sounds can affect a larger area and thus a greater number of species and individuals [19].

Monitoring sound in the ocean has the potential to aid in assessing some of the effects human activities have on marine wildlife, and to better understand the behaviour of several Soniferous species in our oceans [20, 21]. By passively recording sounds from the environment, researchers are able to learn more about marine wildlife and gain a better understanding of how these vocalizing species use sound as a means of communication and observation of their environment. Furthermore, researchers are also able to learn more about how human activities in the ocean affect marine wildlife. Therefore Passive Acoustic Monitoring (PAM) plays an important role in ensuring effective conservation efforts of marine environments to preserve a healthy ocean ecosystem [22, 23].

In recent years the use of unmanned autonomous vehicles in a variety of marine fields has increased [24, 25]. These Autonomous Underwater Vehicles (AUVs), often referred to as *gliders*, enable relatively inexpensive long-term monitoring of marine environments, thus increasing the effective range at which monitoring can be performed. In relation to *acoustic* monitoring of marine environments specifically, these vehicles also have a number of advantages over traditional PAM methods. Most notably they have the ability to move from their originally deployed position. Thus enabling researchers to remotely direct these vehicles to parts of the ocean they deem more suitable for their research. Another important advantage is the fact that these vehicles can be significantly more quiet that other mobile recording platforms, such as ship-towed hydrophone arrays. Ship-towed hydrophone arrays are, as their name suggests, a number of hydrophones connected in series, that are dragged behind a ship. Not only do such recording platforms introduce significant self-induced noise into the collected acoustic data, but they also pose the risk of disturbing the local marine wildlife, thus limiting the applicability of these platforms for Bioacoustic research. Therefore, using gliders as platforms for acoustic monitoring of marine environments could be a valuable tool in ensuring conservation efforts are effective in monitoring the health of ocean ecosystems.

However, the amount of raw data collected during long-term surveys can become too large for human experts to practically sift through, with the purpose of extracting those sound events they deem important to their work for further analysis [22]. Thus, the need to automatically detect and classify sounds within these large audio datasets arises.

Akvaplan-Niva and the GLIDER Project

This master's project was suggested by Akvalplan-NIVA[1], a daughter company of the Norwegian Institute for Water Research, as a part of their ongoing project titled "GLIDER - Unmanned Ocean Exploration" [26]. The project was led by Akvaplan-niva and funded by RNC DEMO 2000 and ConocoPhillips. The project developed and executed the deployment of gliders between Nordland and Troms regions between March and September of 2018 [26]. The passive acoustic data collected in this project showed sound events of both natural and anthropogenic origin. Some of this data has been manually labeled with the containing sound events, thus permitting these to be used as training data for several supervised machine learning techniques.

The goal of this master's project is to explore some of the possibilities machine learning techniques has in aiding researchers working with glider-based marine acoustic data by performing automated sound event detection and classification of the acoustic data collected during the GLIDER project. In this thesis we will refer to this dataset as the GLIDER dataset, and will be introduced in greater detail later in chapter 4.

1.2 Problem Outline

Automated detection and classification of sound events is not a new subject, and has been extensively studied in a variety of contexts. With studies relating to terrestrial environments for a variety of tasks, such as detecting and classifying birds in audio recordings [27], or for environmental sound classification of everyday domestic environments [28]. Sound event detection and classification has also been studied to some degree in marine environments, as recording audio in marine environments similarly is a well established method of data acquisition in several fields. Several of the sound event detection and classification techniques used for marine audio often try to detect the presence of a specific sound source, especially in fields such as marine bioacoustics in which the target sound source often is a specific species [21, 25, 29]. These have utilized various algorithmic

[1]Akvaplan Niva website: `https://www.akvaplan.niva.no/en/home/`

approaches, as will be presented in chapter 3. These techniques often require expert domain knowledge, and specifying their parameters is subject to user error, which in turn can introduce error into the detections themselves. Some examples of such parameters are minimum and maximum frequencies the detector should consider, or the expected call contour of the calls to detect. Furthermore, as many of these detectors can only detect the presence of a *specific* species, or a specific sound event, the recorded audio must be passed through a number of detectors to enable general-purpose sound event detection of the relevant classes for any one research project. As an alternative to such algorithmic detectors, some studies have utilized Machine Learning (ML) based techniques in this domain, as these enable a model to *learn* how to perform the detection and classification task on new data based prior experience, rather than relying on user-specified parameters. Some of these methods split the task of sound event detection and classification into two distinct tasks, first detecting the presence of some relevant sound event, before performing classification of the event to determine the sound source, sometimes using two distinct methods for each task [20, 21]. Meaning that the sound event detection and classification task in these cases are decoupled, rather than utilizing an end-to-end approach.

Some studies in marine acoustic sound event detection and classification have utilized various Deep Learning based Machine Learning models [12, 30–35]. In several of these studies the authors utilize variations of the computer vision related Convolutional Neural Network (CNN) architecture to perform audio classification. As this architecture has provided promising results for a variety of tasks in the computer vision domain, and by converting raw audio data into it's time-frequency representation or similar representations, the transformed data is similar in form to that of images. These CNN based models have arguably been the most common architecture used within the audio domain in general. However, with the introduction of the Vision Transformer (ViT), an adaptation of the successful NLP Transformer model architecture for the computer vision domain, in 2020 by Dosovitskiy *et al.* the transformer achieved state-of-the-art (SOTA) performance for a number of image related tasks [36]. This model was later adapted by Gong *et al.* for the audio domain with the introduction of the Audio Spectrogram Transformer (AST), which achieved SOTA performance on a number of audio benchmark tasks, all of which for terrestrial audio data [14].

A purely attentional model, such as the Audio Spectrogram Transformer (AST), has never been evaluated on *marine* acoustic data, to the best of our knowledge. Neither has any such models been evaluated for acoustic data collected from glider based AUV platforms, as far as we are aware. These platforms, particularly for those platforms that control their buoyancy for propulsion, such as the Seaglider glider design, contain a type of self-induced noise from their internal pumps that is particular to these (or similar) platforms. As far as we are aware, the effect of such self-induced noise from the recording platform on deep-learning based models has not been investigated.

The supervised ML based techniques often utilized in both marine and terrestrial sound event detection and classification, usually require labeled audio data to learn from during training. This, as in several other problem domains encountered in ML, is often one of the biggest difficulties in producing high-performance models. For this reason, alternative machine learning techniques that do not require large amounts of labeled training data such as Unsupervised, semi-supervised or self-supervised techniques, pose an interesting alternative to the supervised techniques for the audio domain, where data volume can be large while the amount of labeled data is comparatively small.

Therefore, exploring the applicability of purely attentional models such as the AST for glider-based marine sound event detection and classification is an interesting endeavour, and worth investigating as these models show promising results in a number of terrestrial audio related tasks [14]. Furthermore, as unsupervised techniques have the advantage compared to supervised techniques, in that they do not require large amounts of manually labeled data for training. An attribute that makes them promising to the marine acoustic domain, where the amount of recorded audio often can be large while the number of labels are comparatively small. Therefore, we compare the performance of unsupervised techniques to supervised techniques, for the final multi-label classification task also worth investigating.

1.3 Research Goals

This thesis aims to explore the possibilities of using state-of-the-art techniques and models from the field of deep-learning, to perform multi-label classification of glider-based marine acoustic data, with the purpose to detect the presence of Biophonic and Anthropogenic sounds in the GLIDER dataset. This is explored through three specific research questions outlined in table 1.1.

RQ1	*How can deep learning models be applied for multi-label classification of anthropogenic and biophonic sound events with the GLIDER dataset?*
RQ2	*How does self-supervised pretraining of the AST (SSAST) affect the performance of multi-label classification of glider based marine acoustic data?*
RQ3	*How does the performance of the CNN-based ResNet18 model compare to the Audio Spectrogram Transformer (AST) and Self-Supervised Audio Spectrogram Transformer (SSAST) for the task of multi-label classification of glider-based marine acoustic data?*

Table 1.1: Research questions

1.4 Preliminary Work

Prior to this project, a preliminary evaluation of a model based on the ResNet18 Convolutional Neural Network (CNN) architecture, on the task of marine audio classification was performed in [1]. In which the ResNet18 architecture was trained and evaluated on the dataset related to the 9th workshop on Detection Classification Localization and Density Estimation (DCLDE) of Marine Mammals using Passive Acoustics [37]. The workshop took place on Waikiki on the Island of Oahu March 7-11 2022. We will refer to this dataset as the DCLDE Oahu dataset. The DCLDE Oahu dataset contains multi-channel hydrophone data collected using ship-towed arrays. The dataset therefore contains self-induced noise in the form of ship engine noise. The CNN-based architecture was selected since CNNs has been common in the field of marine sound event detection and classification [12]. Also because evaluating its performance on the DCLDE Oahu dataset could provide baseline performance that future work using the GLIDER dataset could be compared to.

In our preliminary work [1] the ResNet18 model achieved comparable performance to similar works performing Sound Event Detection (SED) on terrestrial audio. The performance metrics of the ResNet18 model on the DCLDE Oahu dataset from [1] can be found in table 1.2.

Metric	**Moan [1]**	**Other works**
Accuracy	70.0 ± 8.3	63.1 [38]
Precision	79.4 ± 17.0	-
F1 Score	65.6 ± 5.7	69.7 [39]
ROC AUC	79.9 ± 2.1	-

Table 1.2: Performance metrics from table 5.1 of [1] showing the results of the ResNet18 Convolutional Neural Network (CNN) architecture on the DCLDE Oahu dataset compared to other similar works on terrestrial domain.

In this thesis we will evaluate the performance of the AST, the SSAST, and the ResNet18 architecture, for the task of multi-label classification of glider-based marine acoustic data.

1.5 Contributions

There are three main contributions of this work, the first being a quantitative comparison of the performance of three deep learning based models, for multi-label classification to detect and classify anthropogenic and biotic sound events in glider-based marine acoustic data. The second contribution is a thorough understanding of a data processing pipeline that enables us to utilize the GLIDER dataset with deep learning based classification models. With the third contribution being a publicly available code repository[2], containing an implementation of the data processing pipeline, and the trained parameters for the models evaluated in this thesis.

1.6 Thesis Structure

This thesis is structured into six chapters. After the introduction in chapter 1, in chapter 2 the theoretical background related to audio and machine learning is presented. Some of the specifics of marine acoustics are also briefly presented. In chapter 3 work related to marine sound event detection and classification, as well as related work within the field of machine learning, in both the terrestrial and marine audio domain, is outlined to provide an overview of how similar tasks have been approached in the past. In chapter 4 some of the methods used in this work is presented, this will include details about the GLIDER dataset, the implemented pipeline used to train and evaluate the AST and SSAST models, dataset preprocessing steps and other details that are relevant to how the result presented in chapter 5 are created. In chapter 5 the results of the training and testing of the two models used in this work is presented and discussed. Chapter 6 concludes this work by answering the research questions presented in chapter 1, and provides suggestions for future work mainly related to sound event detection and classification of marine glider-based acoustic data specifically.

[2]Github repository: `https://github.com/MartinMoan/TDT4900-Noise-in-the-Sea-Source`

Chapter 2

Background

In this chapter the theoretical background that is required or otherwise useful when working in the audio domain is introduced. The chapter first introduces some of the concepts related to audio and signal processing in general, after which the models used and evaluated in this work is presented and their architectures explained. Thereafter the chapter goes on to explain some of the topics that relate to machine learning in the audio domain specifically.

Much of this chapter, specifically the parts of section 2.1 are heavily influenced by our previous work [1], due to much of the theoretical background relating to sound being the same for both this project and [1].

2.1 Audio

In this section we will introduce and explain some of the terminology, methods and other topics related to the audio domain.

2.1.1 Sound and the Waveform

Sound is defined as a physical disturbance that moves through an elastic medium [40, 41]. These disturbances propagate through the medium as series high and low pressure, also known as compressions and rarefactions, which decrease in amplitude with distance from the source.

We can record sound by measuring the variation in pressure of the surrounding medium over time. Such recording can be analog, with the pressure being recorded as a continuous signal, or digitally with discrete values sampled at some fixed time interval, called the *sampling rate* of the signal. For the remainder of this thesis, we will refer to digital audio signals specifically when discussing audio. In air we can record sound using microphones, devices that most of us likely have some experience with using in our electronic devices. In aquatic environments we can record sound by using specialized underwater microphones called *hydrophones*. These are acoustic recording devices that are adapted to the impedance of water as opposed to that of air, to enable them to record sound in water more effectively than normal microphones [42].

If we plot the variation in pressure, or amount of disturbance of the medium, as a function of time, we get a graph called the *waveform*. An example of this is shown in figure 2.1

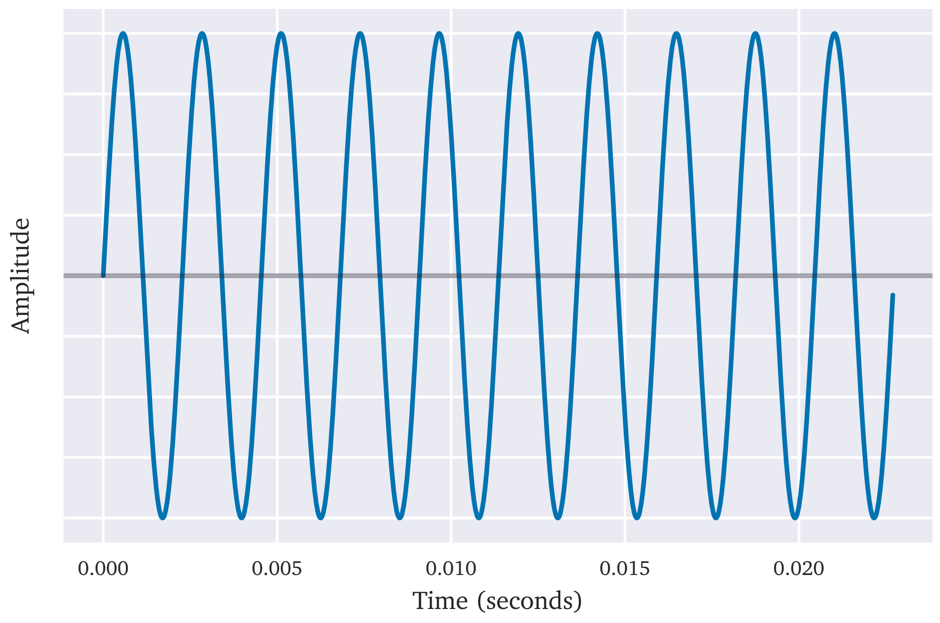

Figure 2.1: Concert A (440Hz) waveform, figure 2.1 from [1]

The signal displayed in figure 2.1 shows a graphical representation of a sound signal as a continuous waveform. However, when working with digital audio this representation does not describe the recording accurately. With digital audio data, a recording consists of a series of discrete numeric values, called *samples*, recorded at discrete time intervals. The set of unique discrete values any sample may take is defined by the *bit depth* of the recording, meaning the number of bits used to represent an individual sample. The number of samples recorded per second is called the *sampling rate* of the signal. A digital waveform therefore does not consist of a continuous wave, but is more accurately represented by a scatter plot. A basic example of how the sampling rate breaks up the continuous signal into individual discrete samples can be found in figure 2.2.

The waveform in figure 2.2 consists of a generated sine wave with constant frequency of 440 Hz and constant amplitude. For real audio however, which often contains sounds from several sources, at varying amplitudes and frequencies, the waveform will usually look more erratic. An example of the waveform of a real audio recording can be found in figure 2.3, in which we see that the waveform varies more than the constant waveform.

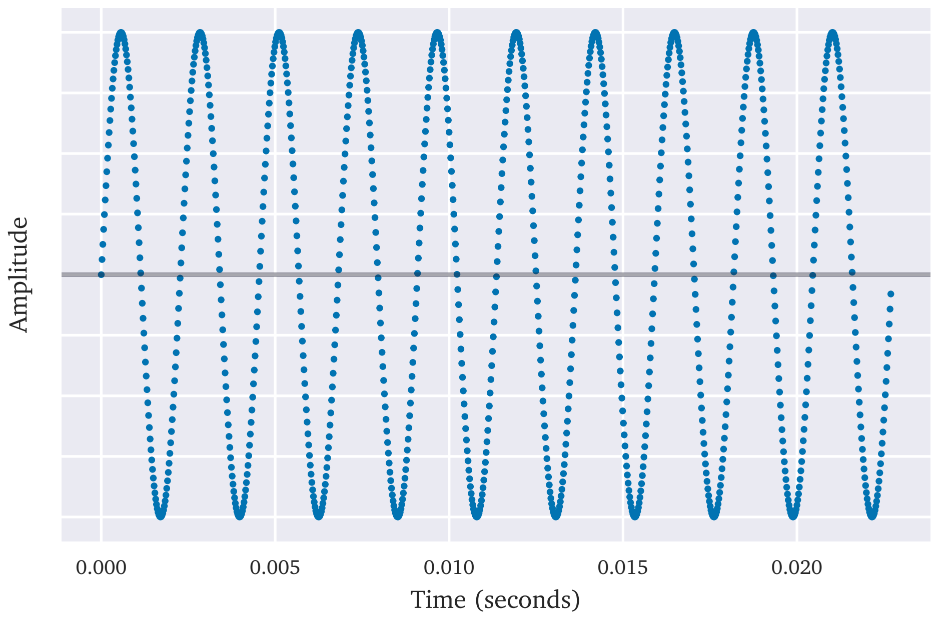

Figure 2.2: Concert A (440Hz) sampled at 44100Hz, figure 2.2 from [1]

The Nyquist–Shannon sampling theorem

The Nyquist–Shannon sampling theorem describes a fundamental limit related to the sampling of any continuous time signal [43]. The sampling theorem tells us that the maximum frequency f_{max} we are able to perfectly reconstruct, meaning without introducing undesirable artefacts, by sampling when using a sampling rate S_r is defined by the following relationship:

$$f_{max} = \frac{S_r}{2} \tag{2.1}$$

This is also known as the *Nyquist frequency* [43]. Conversely, if we know the maximum frequency f_{max} we need to reconstruct by sampling, we know that the minimum sampling rate we need to utilize is $2f_{max}$ [43]. The sampling theorem was, indirectly, used to determine the sampling rate of some digital audio formats intended for entertainment, such as for Compact Disks (CDs) [44]. Empirical results of the frequency spectra audible to humans show that most humans can hear frequencies up to about 20 kHz, although this varies from person to person, and as our hearing degrades with age this maximum frequency usually decreases [45, Chapter 13]. Therefore the 44.1 kHz was selected as the sampling rate used for CDs, ensuring that most frequencies audible to humans could be recorded completely with no artefacts in the recordings [44].

If the sampling rate of a signal containing a maximum frequency f is sampled at a rate S_r lower than $2f$, the recording will exhibit a type of artefact called

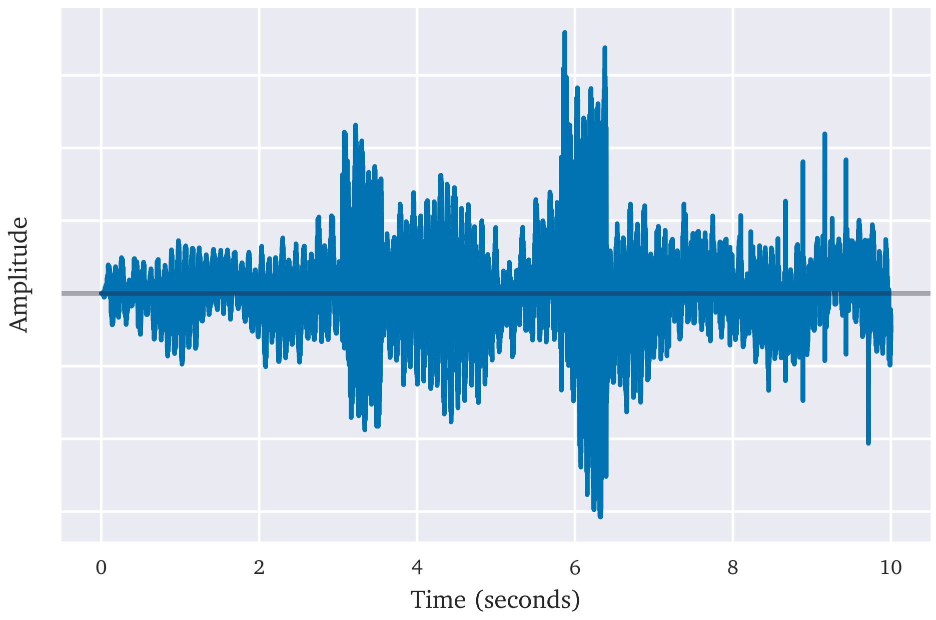

Figure 2.3: Audio waveform sampled at 44100Hz, figure 2.3 from [1]

aliasing [43]. An example of aliasing due to undersampling can be found in figure 2.4. In this figure we see that the sampling rate used to record the underlying signal is not sufficient to determine the frequency of the signal. To clarify this point, consider a case in which the frequency of the signal were doubled, but the sampling rate remain the same. In this case the values of the samples we record would remain the same, even though the underlying signal is significantly changed. This effect is called Aliasing and is an artefact of insufficiently sampled signals. To avoid this a low-pass filter is usually applied to the signal before sampling, thus eliminating frequencies above the Nyquist frequency from the signal before sampling [43].

The Rayleigh Frequency

The Rayleigh frequency defines the minimum frequency F_{min} in Hertz that can be reconstructed with a sampled signal of T seconds as $F_{min} = \frac{1}{T}$ [46]. This means that when computing the spectrogram of a sampled signal, using the STFT in which we compute the Fourier transform of successive windows of a signal, the duration of this window determines the minimum frequency we can effectively resolve in the spectrogram.

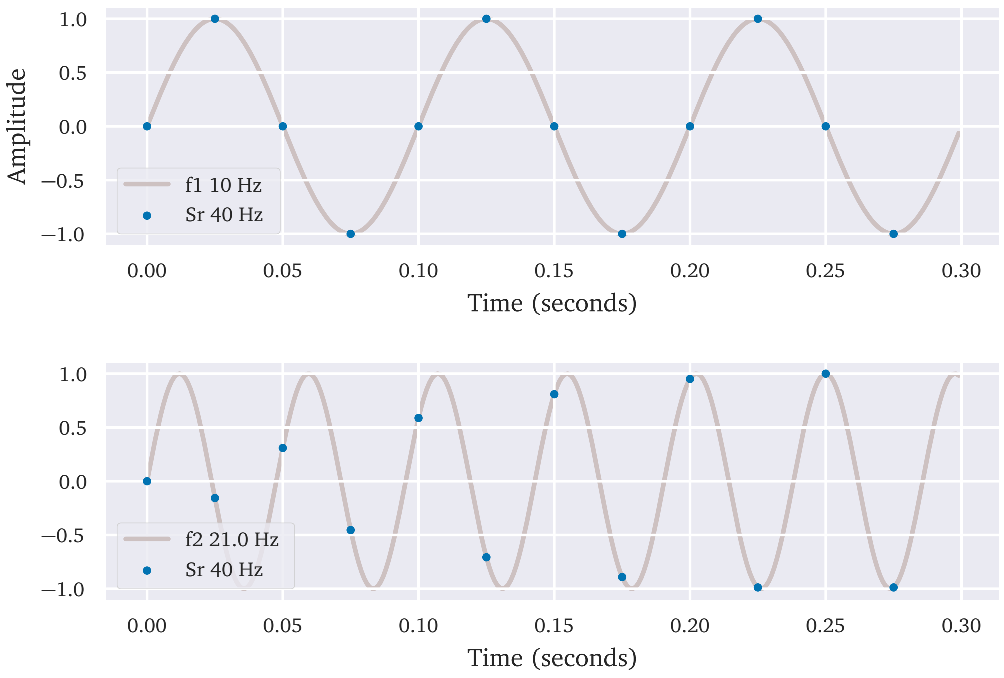

Figure 2.4: Example of aliasing due to undersampling, figure 2.8 from [1]

2.1.2 The Fourier Transform

In natural environments, the soundwaves from different sources are combined as the sum of the individual soundwaves at a point in the environment. Meaning that the instantaneous pressure measured at a recording device is equal to the sum of the individual waveforms as they coincide with the recording device. This means that the individual waveforms from any two sound sources can form constructive or destructive interference when their individual signals reach the recorder. However, even though it is apparent that a waveform can consist of the sum of (possibly) several frequencies, it is not clearly visually apparent from the waveform *which* frequencies are present within the recorded signal. Often when working with audio data we want to analyse the frequency content of some signal, such that we can better understand which frequencies contribute in creating the recorded waveform.

For this we can use the Fourier Transform to transform the waveform from the time domain to the frequency domain [47]. In doing so we can inspect the individual frequencies that contribute to the waveform. The Fourier transform exists in several varieties, both for continuous signals and discrete signals. For the purposes of this project we will only consider the *discrete* Fourier transform, as this version of the transform enables us to decompose discrete audio signals into their frequency components, which is the form all audio data used in this project will take. The discrete Fourier transform as described by [48], of a discrete signal containing N samples is defined by equation 2.2:

$$X(k) = \sum_{n=0}^{N-1} x(n) \exp(-i\frac{2\pi}{N}kn), \quad k = 0, \ldots, N-1 \tag{2.2}$$

Where k is the k-th frequency component of the signal. In equation 2.2 the term $\exp(-i\frac{2\pi}{N}kn)$ is a complex number written in exponential form. Therefore the resulting sum will also be a complex number. As we are only interested in the real analog frequencies within the signal, we can account for the complex symmetry that is present in the DFT in equation 2.2. This involves computing the same DFT for $k \leq \frac{N}{2}$. To ensure that the output amplitudes of the DFT components are correct, we average the magnitude of every component by multiplying its magnitude by $\frac{2}{N}$. The magnitude of a complex number is defined as [49]:

$$|e^{ix}| = |\cos x + i \sin x| = \sqrt{\cos x^2 + \sin x^2}, \quad i = \sqrt{-1} \tag{2.3}$$

So for a real valued signal S consisting of N sampled values, discretized with a sampling rate of S_r, we can compute the magnitude of the real valued frequencies of the signal bellow the Nyquist frequency by:

$$X(k) = \frac{2}{N} \left| \sum_{n=0}^{N-1} x(n) \exp(-i\frac{2\pi}{N}kn) \right|, \quad k = 0, \ldots, N/2 \tag{2.4}$$

Where the analog frequency in Hertz corresponding to the k-th DFT component is, according to [48], defined as:

$$f_k = k\frac{S_r}{N} \tag{2.5}$$

An example of this DFT computed for a generated 5 second sinewave of 5 Hz with amplitude of 1 (with no unit) sampled at 80 Hz can be found in figure 2.5. In figure 2.5 we see that the spectrum is only defined for frequencies below the Nyquist frequency $\frac{80\,\mathrm{Hz}}{2} = 40\,\mathrm{Hz}$.

The Fourier transform can be a valuable tool for signals whose frequency content is stable, but if the signal consists of varying frequencies or varying amplitude, the Fourier transform will not help us understand how the contents of the signal changes over time. For most sounds we might encounter in natural environments, the contributing frequencies usually vary. If we need to inspect how the frequencies, and their amplitudes, of these sounds change over time, we can use the Fourier transform to convert the signal into it's time-frequency representation and plot it using a Spectrogram.

2.1.3 The Spectrogram - Short-Time Fourier Transform

The spectrogram is a time-frequency representation of, usually, a time-domain signal such as an audio waveform. The spectrogram enables us to see how the

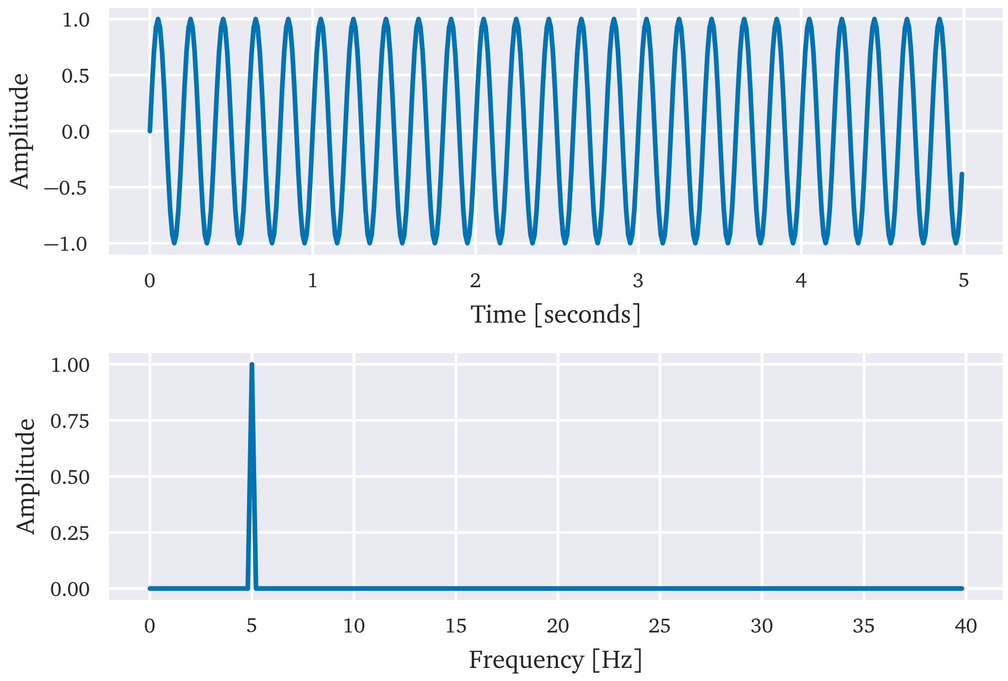

Figure 2.5: Example of real valued Discrete Fourier Transform (DFT) (bottom) for generated 5 Hz sinewave (top) with unit amplitude sampled at 80 Hz

frequency content of a signal changes over time, which can be useful for audio originating from natural environments, where the frequency content of sounds typically vary with time. The spectrogram can be computed in a number of ways with a number of transformations, but arguably the most common method used in the audio domain is the Short-Time Fourier Transform (STFT).

The Short-Time Fourier Transform

The STFT is an algorithm in which, given some time-domain signal, we perform the Fourier transforms of successive *windows* of the signal. Resulting in frequency spectra we "stack", thus enabling us to visualize how the frequency spectra change over time. Meaning that we split the signal up into smaller windows, compute the Fourier transform on the samples within each window, shift the window forward some number of samples, and repeat this process. We do this successively until the we have traversed the entire input signal. An example of a typical time-frequency representation spectrogram can be found in figure 2.6.

The horizontal axis of the spectrogram in figure 2.6 represents time, and the vertical axis represents frequencies. One thing to note is that the vertical axis of the spectrogram does not represent single specific frequencies, but rather a range, or *band*, of frequencies. The width of these bands W_{band} is defined by the length

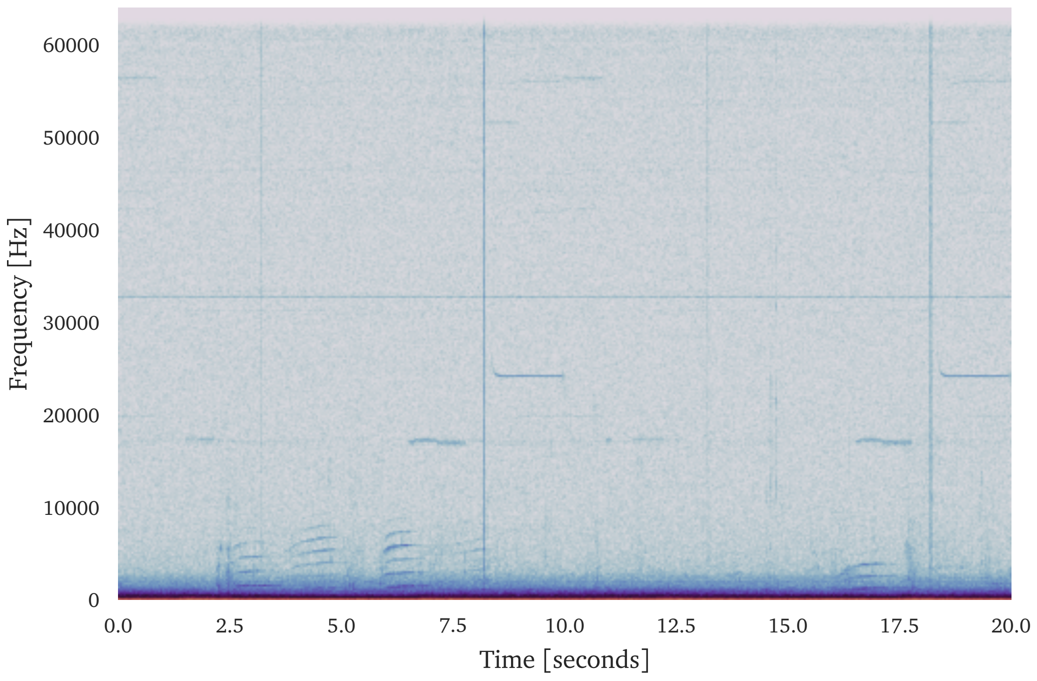

Figure 2.6: Spectrogram of an audio clip of the GLIDER dataset

$N_{dft}$of the Fourier transform window used when performing the STFTs and the sampling rate of the original signal by equation 2.6.

$$W_{band} = \frac{S_r}{N_{dft}} \tag{2.6}$$

Given a signal with sample rate S_r of, for example, 44.1 kHz and using a STFT window length of 516 samples, the result of each Fourier transform would be a vector where each of the vector components would represent a range of frequencies W_{band} spanning $W_{band} = \frac{44.1\,\text{kHz}}{516} \approx 85.5\,\text{Hz}$ frequencies. The windows of the STFT typically overlap by some number of samples, and are often scaled at its terminals using a windowing function to avoid spectral leakage, an effect where frequency components of the Fourier transform seem to *leak* into adjacent frequencies [50]. In this project we use the Hann windowing function [51] as this is the default windowing function applied by the librosa [52] Python [53] library, and as we found no significant improvement in spectrogram quality with other windowing functions through informal experiments.

2.1.4 The Mel Scale - Frequency and Pitch

The term *frequency* in relation to sound, describes the number of oscillations within the propagation medium that occurs per second, expressed in *hertz*. The term Pitch however, refers to how we, as humans, *perceive* frequency. As humans we typically perceive the distance between two low frequencies as greater than for two higher frequencies [6]. An Octave describes the interval between two frequencies f_1 and f_2, where one frequency is twice as large as the other [8]. We typically perceive such an interval as consisting of two versions of the same tone, with one being a higher pitch version of the other [8]. For example, consider the tones concert A1, A2 and A3. Where A2 is one octave above A1, and A3 one octave above A2. Their specific frequencies being $A1 = 440\,\text{Hz}$, $A2 = 2 \cdot 440\,\text{Hz} = 880\,\text{Hz}$ and $A2 = 2 \cdot 880\,\text{Hz} = 1760\,\text{Hz}$. If we calculate the change in frequency between successive octaves, we find that the change in frequency from A1 to A2 is $880\,\text{Hz} - 440\,\text{Hz} = 440\,\text{Hz}$, and the change in frequency between A3 and A2 is $1760\,\text{Hz} - 880\,\text{Hz} = 880\,\text{Hz}$. From this we can tell that even tough we perceive the interval between A1 and A2 to equal the interval between A2 and A3, the difference in physical frequency between these two intervals is substantially different. This has led to the development of the Mel scale (short for melodic), which tries to map intervals in physical frequencies to how we perceive these intervals [6].

There is no single "correct" mel-scale, as these are typically found empirically, through experiments involving human listeners [6]. However, the mel scales tell us that humans do not perceive pitch linearly. The mathematical expression of one mel scale that is often used can be found in 2.7, which enables the conversion of a physical frequency f expressed in hertz to be converted into its mel scale equivalent m [6].

$$m = 2595 \cdot \log_{10}\left(1 + \frac{f}{700}\right) \tag{2.7}$$

Work relating to deep-learning in the audio domain often utilize mel scaled features, especially for tasks relating to speech recognition, as several of these models seem to perform better with mel scaled features than normal frequency spectrograms [54, 55].

2.1.5 Describing Sound Levels

When working with audio data, we should be aware of how sound levels can be described. For this the decibel (dB) scale is often used. The decibel scale is a relative unit that describes one tenth of a *Bel*. The scale is *relative* as it describes the *ratio* of two quantities, such as power, pressure or volts [56]. The decibel is described as ten times the logarithm of the ratio of a quantity S to some reference quantity R [56], expressed mathematically as:

$$dB = 10 \log_{10}\left(\frac{S}{R}\right) \tag{2.8}$$

When an object in a medium vibrates, it introduces a vibration into the surrounding medium, thus performing an amount of work on the medium. This work, or the amount of energy, introduced by the object over time is called the *sound* *power*, and is measured in Watts [56]. This quantity can also be described by using the decibel scale of the sound power relative to some reference power, called the *sound power level*. For sounds in air the reference power used is sometimes the sound power at the threshold of human hearing of approximately 1×10^{-12} W [56]. The sound power *level* of a sound source with sound power S is thereby described with this reference power in decibels as [56]:

$$\text{Sound Power Level (dB)} = 10\log_{10}\left(\frac{S}{1\,\text{pW}}\right) \tag{2.9}$$

The sound power tells us the amount of energy introduced by the object to the medium at the sound source. The vibrations induced in the medium by the object will propagate through the environment as a series of compressions and rarefactions of the medium. For gasses and liquids we can measure these compressions and rarefactions as the variation of pressure from ambient pressure at a point in the environment some distance from the source. Thereby describing the sound pressure of the sound in Pascals. The sound pressure describes the amplitude of the deviation from ambient pressure as the soundwave propagates over the measuring device. We can also describe this quantity in decibels called the *Sound Pressure Level (SPL)*, using a reference pressure of 20 µPa typical for sound in air [57]. We can compute the sound pressure level in decibels of a measured sound with pressure deviation of P Pascals by [56]:

$$\text{Sound Pressure Level (dB)} = 10\log_{10}\left(\frac{P}{20\,\mu\text{Pa}}\right) \tag{2.10}$$

Sound *intensity* is the average amount of energy that passes through an area over time [58]. Because power describes an amount of energy per unit time, sound intensity can therefore be considered as the flux of *energy* that passes through an area per second. Therefore we can describe the intensity of some sound as the power, that passes through an area, in Watts per square meter. We can describe the sound intensity in decibels as the *sound intensity level*, using a reference intensity approximating the threshold intensity of human hearing in air of $1\,\text{pW/m}^2$ [57]. The Sound Intensity Level (SIL) of a sound with an intensity of $I\ W/m^2$ can be described in decibels with the above reference intensity as [56]:

$$\text{Sound Intensity Level (dB)} = 10\log_{10}\left(\frac{I}{1\,\text{pW/m}^2}\right) \tag{2.11}$$

The intensity of a sound wave is related to the pressure of the wave P, the density of the propagation medium ρ and the speed of sound in the medium c [57]:

$$I = \frac{P^2}{\rho \cdot c} \tag{2.12}$$

Using equation 2.12, we can rewrite the equation for sound intensity level from equation 2.11, such that it is only dependant on sound pressure and reference pressure using the following equations [57]:

$$I = 10\log_{10}\left(\frac{I_{sound}}{I_{reference}}\right) \quad (2.13)$$

$$= 10\log_{10}\left(\frac{P^2_{sound}}{P^2_{reference}}\right) \quad (2.14)$$

$$= 20\log_{10}\left(\frac{P_{sound}}{P_{reference}}\right) \quad (2.15)$$

With this equation we can describe the sound intensity level in decibels based on the sound pressure measured by a recording device and a reference pressure. However, the reference pressure used is typically different in water and air. Such that sound intensity level of 10dB in air does not entail the same intensity as 10dB in water. The reference pressure for air is commonly 20 µPa whereas the reference level for water is usually 1 µPa [57].

For the purposes of this thesis, the sound intensity level is not directly relevant, as we do not have to describe the intensity of sound explicitly to the models we will use. Because neural networks such as the ones we will use, typically perform better if their inputs are normalized to have zero mean and unit variance. Such that after normalization the actual value of the sound intensity level described with the decibel scale would be be irrelevant. However, as we will see later in chapter 4 we can still utilize the decibel scale when normalizing input data due to the logarithm applied by the decibel scale.

2.1.6 Sound in Marine Environments

So far some of the theoretical background related to sound and audio signal processing has been presented. However, the specifics of sound and audio in marine environments has not yet been elucidated.

Water is a far more effective medium for acoustic propagation than air. The speed of sound in air is approximately 343 m/s, while the speed of sound in water is approximately 1490 m/s [59]. The speed of sound in water is affected by a number of factors such as temperature, salinity and pressure [59]. At the depth where the sound speed is at its minimum, around 1000 m, there exists an acoustic property called the Sound Fixing and Ranging channel (SOFAR) channel, or Deep Sound Channel (DSC) [60, 61]. In this channels an effect known as waveguiding typically occurs for soundwaves produced within the channels. Waveguiding in marine acoustics is an effect where soundwaves are trapped within a channel, "boucing" between the surfaces of the channel through refraction. This causes the sound to propagate horizontally within this waveguide for distances of several hundred kilometers [61, 62]. Furthermore, as sounds can bounce off both the ocean surface and the seabed, sound in marine environments can take multiple paths when traveling to a target, an effect known as multipath sound propagation [63, 64]. This effect can cause the sound at each individual path to arrive at different times at the target [65]. Multipath sound propagation is not exclusive to marine environments, but can be more significant with marine audio due to the range at which sound can travel in marine environments, compared to the same effect in terrestrial environments.

These effects further complicates the task of sound event detection and classification using acoustic data collected with *mobile* marine recording platforms, such as AUV gliders, as these platforms traverse the watercolumn. As the AUV traverse the watercolumn, the salinity, temperature and pressure around the recording platform changes, thus changing the acoustic properties of the recording environment.

2.2 Machine Learning

In this section some of the theoretical background related to machine learning that is relevant in understanding the models used in this work is presented. This will include a brief overview of the transformer model as it was originally introduced in [13], some of the details regarding how it was adapted for the vision domain in [36], as understanding these aspects in important in understanding how the AST (and SSAST) models work. This section will also introduce some of the background theory relevant to machine learning in general.

2.2.1 Formulating the optimization task

It is important to clarify a few concepts related to sound event detection and classification and how it relates to audio classification and multi-label classification of audio data.

In the field of machine learning, classification is the task of predicting the class an example datapoint corresponds to. If the number of possible classes in the dataset is more than one, the task would be considered *multiclass classification*. If there is only one possible class in the dataset however, the task would be described as *binary classification*. Where a model would attempt to classify whether the input belongs to the one class or not. In both of these cases the example can belong to one, and only one, of the possible classes. Some examples of such tasks can be the binary classification task to predict whether a patient has cancer or not based on their medical history and symptoms. Or the multiclass classification task to classify images of birds according to species. In these cases an example can only correspond to *one* of the possible classes.

In multi-label classification however, the input example can belong to more than one class. Examples of multi-label classification tasks can be to detect the presence of cats, dogs and parrots in an input image. In this case the image may indeed contain just a dog, or a cat or a parrot, or all of the classes simultaneously. We can also consider this task as performing multiple independent binary classification tasks, detecting the presence of each of the three classes independently from the other classes.

The task of sound event detection and classification is not new, and has been studied extensively in the past. The task, as its name suggests, includes detecting whether some sound event has occurred in some audio, and if so predict what class or classes the sound event corresponds to. The detection part of the task therefore involves to determine *when* in a recording the sound event occurs. The temporal resolution of this detection that is required depends on the usage of the final model predictions. For example, if we try to detect the presence of transient sounds in a recording, such as a car horn, a dog bark, or speech, the temporal resolution required in the detection will likely be much finer than the temporal resolution required to detect more constant sounds, such as the presence of engine noise from a ship in marine audio. Depending on the number of target classes, sound event detection and classification can involve binary or multi-label classification. Considering that multiple sound events can overlap temporally it is clear that the task is better described as multi-label rather than multiclass. As we will be presented in greater detail in chapter 3, in several cases work related to sound event detection and classification perform detection and classification as distinctly separate tasks, sometimes with differing models and methods for the detection and classification parts of the pipeline.

In the marine audio domain, the raw audio recordings can last for hours, with successive recordings thus spanning days, weeks or months. While the sound events researchers typically try to investigate from these recordings are, comparatively, significantly shorter. The whistle of an odontocete for example, such as the orca, typically lasts just a few seconds. There are a multitude of methods we can employ to detect and classify these sound events. One method is to perform multi-label classification of entire recordings, referred to as classification at the recording level. In this case a (true) positive prediction would tell us that the input recording contains a sound event by one of the target classes. However, because the sound events we are interested in detecting often only span a small portion of the recording, this does not help us in predicting where within the recording the event occurs. Another option is to split the recording into shorter segments, or *clips*, and perform the same multi-label classification for each individual clip. A (true) positive prediction for the i-th clip would thereby enable us to predict roughly where within the original recording the sound event occurs, based on the duration optional overlap of the clips. In both cases we are able to perform joint detection and classification with a single forward pass of the neural network model.

Another option to perform similar joint sound event detection and classification, is to alter the model output and target labels in such a way that they represent, for example, where in the input (recording or clip) a sound event onset and offset occurs. This is called a *event roll* [55]. An example of a spectrogram with its event roll can be found in figure 2.7.

The event roll has the advantage that it enables us to train models that predict the onset and offsets of sound events in the input with significantly better temporal resolution, typically on the scale of individual frames of the input spectrogram.

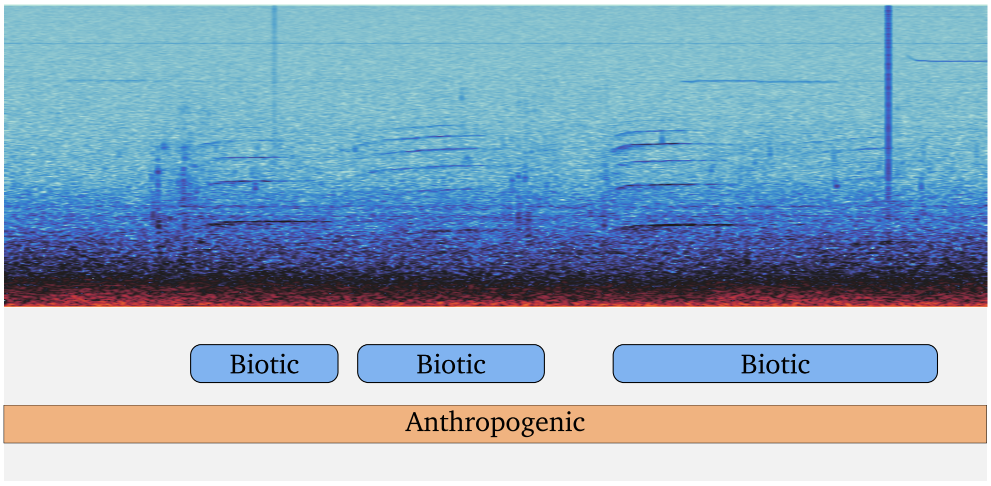

Figure 2.7: A spectrogram (top) with an example event roll (bottom)

However, this advantage depends on the temporal resolution of the available labels. This is referred to as the *strength* of the dataset labels [55]. Where weak labels are labels that do not define the sound event onset and offsets accurately [55]. For example if such labels only specify which recordings in a dataset a sound event occurs in, rather than the onset and offset of the sound event within that recording, the labels are typically considered weak. Whereas strong labels tell us where within a single recording a sound event starts and ends. If our dataset is weakly labeled, and we train our model to predict the event roll, the rows of the event roll would be positive for the entirety of the recording, for most recordings. Therefore, if our dataset is weakly labeled and we train our model to predict the event roll, we would only have introduced computational overhead due to the the increased output dimension, while not gaining any advantage in terms of temporal resolution in the output predictions. Figure 2.8 shows how weak and strong labels can are represented in an event roll. In the top plot we see an example of the event roll with strongly labeled data, where we see that the labels span just parts of the underlying recording or clip. In the top plot we can clearly tell exactly where the sound events start and end. In the bottom plot however, we see how weak labels span the entirety of the underlying recording or clip. Considering that some sounds might not be transient, a label might indeed be considered strong even if it does span an entire, or multiple, recordings or clips. An example of this might be propeller noise from a passing ship, in which case the sound may be present in the recording(s) for a long time as it passes. In this case the labeling for the propeller noise could indeed span multiple recordings, and still be considered strong if it accurately describe the onset and offset of the sound. However, this raises the question of how to decide when a sound event occurs, as a sound event may not necessarily have distinct start and ends, but rather increase and decrease in volume continuously. These are issues that should be considered when labeling

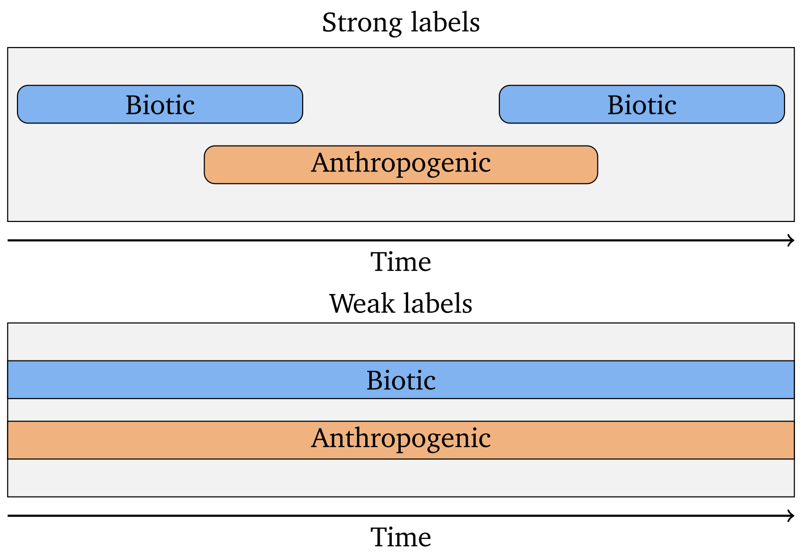

Figure 2.8: Example of weak and strong labels in event rolls, inspired by figure 4 from [55]. Weak labels can span an entire recording or clip, whereas strong labels describe sound event onsets and offsets within the recording or clip.

is performed, by considering the temporal scales of the target sound events.

As we will see later in chapter 4, the labels of the GLIDER dataset have a mean duration of 28 minutes 30.39 seconds with a standard deviation of 15 minutes 39.69 seconds. Considering that the files of the GLIDER dataset typically are about 10 minutes in duration, this means that many of the labels of the dataset span at least the entirety of a single recording. Furthermore, the sound events that the labels describe include biotic vocalizations, which typically are transient in nature. From this it is clear that the labels of the GLIDER dataset are not strong enough to warrant using the event roll as the target and prediction format. Therefore, in this work, the final prediction task we evaluate is multi-label classification of the audio clips. Such that we may gain some of the ability to discern the temporal location of sound events within a recording by multi-label classification of individual clips, while avoiding the unnecessary computational overhead of constant event rolls.

2.2.2 The Transformer Architecture

The original Transformer architecture introduced by Vaswani *et al.* in 2017, was a sequence transduction model [13] intended for Natural Language Processing (NLP) tasks. Sequence transduction tasks are tasks where the model should map an input sequence to an output sequence. An example of this is machine translation where, for example, an input sentence in English would be presented do the model, whereby the model should output the translated sentence in French. The Transformer tried to improve upon contemporary sequence models, such as the

Recurrent Neural Network (RNN) and Convolutional Recurrent Neural Network (CRNN), which all generate their outputs sequentially. This sequential operation of contemporary sequence models rendered parallel computation in the forward pass of the model impossible [13]. The Transformer architecture alleviates this problem, by enabling fully parallel computation in the forward pass of the model during training by solely relying on the self-attention mechanism, which will be presented later in this section, thus improving upon training time [13].

Similar to many of the contemporary sequence models, the Transformer utilizes encoder and decoder sub-networks, where the encoder network enables the Transformer to transform the sequence of input symbols (words of the input sentence) into an internal sequence of continually valued representations. The general Transformer architecture can be found in figure 2.9 below. In figure 2.9, we see the encoder and decoder networks in the left and right parts of the figure respectively. After the internal representation of the input sequence is created by the encoder network, it is then passed to the decoder network. The decoder network then transforms the internal representation sequence into output tokens, for the task for English-French machine translation, the output tokens correspond to the words of the output sentence in French, assuming the model is accurate. The decoder network produces this output auto-regressively [13], meaning that the output at position i is produced the model output at positions before i. This is similar to recurrent models which create their output at position i based on the previous hidden state for input at position $i-1$ [66].

However, if the decoder output was computed auto-regressively during training, we would have to compute the output tokens in sequence, such that we could provide the previous output decoder token as input to the decoder. This would be very similar to the sequential computation of traditional recurrent architectures, and we would risk that error in the previous token(s) would accumulate to the next tokens, and we would loose the ability to compute the output in parallel, thus making training less efficient. Therefore, for sequence transduction tasks such as machine translation, a masking operation is applied to the "regressive" decoder input tokens ("Output embeddings" in figure 2.9) matrix before passing it to the decoder network, which facilitates parallel computation during training [13]. This ensures that the decoder network is only ever able to attend to previous output/target tokens, and the entire sequence representation from the encoder network, when computing the output of the current token, while also enabling parallel computation of the entire output decoder sequence. Thus simulating auto-regressive computation of the decoder output during training. While during inference, the decoder network is in fact run auto-regressively, computing the next output decoder token based on the entire encoder sequence representation and the previous decoder output token, because we do not have access to the target output tokens during inference as we do during training.

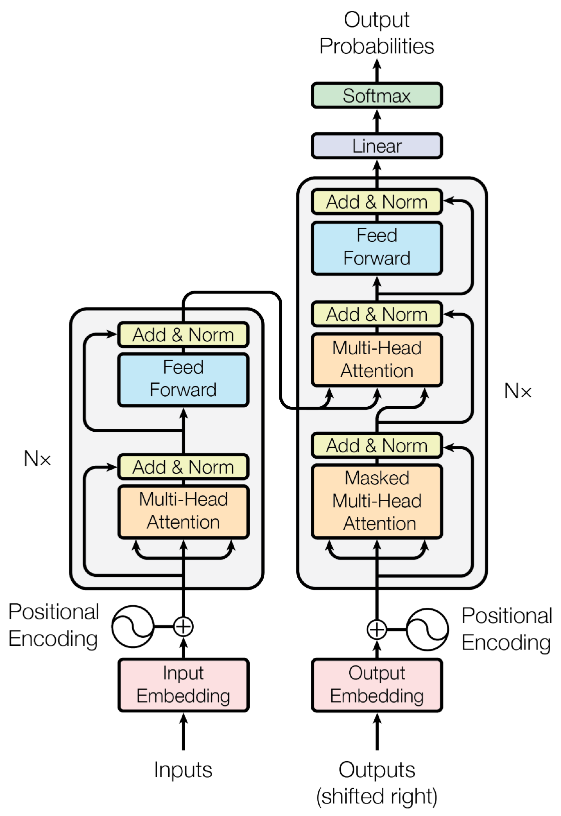

Figure 2.9: The Transformer architecture, Figure 1 from [13]

The Self-Attention Mechanism

The original Transformer architecture utilizes a self-attention mechanism the authors call *scaled dot-product self-attention* [13]. We will refer to this simply as attention or self-attention for the remainder of this thesis. The self-attention mechanism, given an input sequence of tokens, produces a sequence of outputs with the same length as its input. The self-attention mechanism enables each element of the sequence input to *attend* all other elements in the sequence, regardless of the number of elements separating the two elements. This enables the Transformer model to, technically, learn arbitrarily long-range dependencies within the input. While in practice the dimensionality of the input is still limited by computational resources available during training and in deployed environments, thereby limiting the practical dependency range discernible to the model. According to [13], the self-attention mechanism is defined by:

$$\text{Attention}(Q,K,V) = \text{softmax}_{row}(\frac{QK^T}{\sqrt{d_k}})V \tag{2.16}$$

Where the query Q, key K and value V matrices are be created by linear projections using individual learnable weight and bias parameters, each computed on the same input sequence [13, 67]. For the encoder network, given a real-valued input matrix $I \in \mathbb{R}^{n,d}$, consisting of n row-vectors with d elements, each row representing a single input token, the Q, K and V matrices are all created from individual linear projections of I. For the encoder-decoder layers in the decoder network however, the Q and V comes from the output of the encoder, while the K matrix comes from the output of the previous decoder layer [13].

The softmax operation is described in equation 2.17 below [68].

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{2.17}$$

It is necessary to mention that as the self-attention mechanism does not maintain any information as to the *order* of the input embeddings, the authors of [13] inject positional information into the word embeddings by adding a positional encoding PE to the word embeddings, before passing them to the encoder and decoder networks. In [13] the authors use sine and cosine positional encoding, defined by equations 2.18 and 2.19 below. While using learnable positional are also alternatives that are used in the literature [13–15, 36].

$$PE_{(pos,2i)} = \sin\left(pos/10000^{2i/d_{model}}\right) \tag{2.18}$$

$$PE_{(pos,2i+1)} = \cos\left(pos/10000^{2i/d_{model}}\right) \tag{2.19}$$

Where pos corresponds to the position of the embedding in the input sequence, while i represents the i-th dimension of the embedding vector [13]. By adding this positional information to the input embeddings, the Transformer can utilize the order of the input in the prediction task [13].

This outlines the basic function of the Transformer architecture and the self-attention mechanism. In the following sections the Vision Transformer (ViT) and the Audio Spectrogram Transformer (AST) will be outlined. These models are, as their name suggest, based on the Transformer architecture, but with some notable alterations, which will be discussed in the following sections.

2.2.3 Transformers for the Vision and Audio Domain

The AST was introduced by Gong *et al.* in 2021 in [14], and provided state-of-the-art (SOTA) results on a variety of tasks in the audio domain, which will be presented later in chapter 3. The AST is based on the Vision Transformer (ViT) by [36], in which the Transformer architecture that was originally intended for Natural Language Processing (NLP) related tasks, was adapted for the computer vision domain.

The Vision Transformer (ViT)

The Vision Transformer introduced a number of alterations to the Transformer model, which enabled it to be used in the computer vision domain [36]. In the sequence transduction tasks that the Transformer architecture typically was used for, both the input and target values are sequences of vector embeddings. Therefore, to adapt the Transformer to be able to use it with image data, the authors of [36] split the input images into *patches* of 16x16 pixels, and linearly projected the flattened patches using a learnable linear projection layer. Thereby converting the input image with shape $HxWxC$ (height, width and 3 color channels) into a sequence of $PxPxC$ embedding vectors, called *patch embeddings*, which could then be passed to the Transformer model [36] as a proxy for word embeddings.

As previously mentioned, the original Transformer architecture utilizes encoder and decoder sub-networks, where the encoder network enables the Transformer to transform a sequence of input symbols (words of the input sentence) into and internal sequence of continuous representations [13], which can be passed to the auto-regressive decoder network. The ViT forgoes of the decoder part of the original Transformer architecture, as the ViT was not trained as a sequence-to-sequence model but was to perform image classification. Meaning that because the ViT is trained for image classification, it does not produce a sequence of outputs given a sequence of inputs. But rather, given a sequence of image patch embeddings, produce a single classification for the entire input sequence. To facilitate using the encoder for classification, the ViT prepends the typical $[CLS]$ token to the sequence of projections, and use the output of this token for classification purposes in the same manner as was done in [69].

The Audio Spectrogram Transformer (AST)

In [14] the authors adapt the ViT for the audio domain, by applying the ViT to time-frequency representations of the audio data. The most notable change the authors of the AST make to the original ViT architecture is to in regard to the three RGB color channels of typical digital images. As time-frequency representations of sound does not have multiple color channels like what colored digital images have, the authors adapt the ViT by averaging the three color channel weights of the ViT linear patch projections. Thus enabling the ViT to be applied to audio spectrograms.

The Self-Supervised Audio Spectrogram Transformer (SSAST)

In [15] the authors of the original AST paper adapt the model to perform self-supervised pretraining on audio data. The authors adapt the AST to perform self-supervised joint discriminative and generative modeling of masked spectrogram patches [15]. Meaning that after the time-frequency representation of an audio signal is split into patches, such as for the supervised AST, a random set of patches are masked by setting their values to a learnable mask embedding [15]. A positional embedding is then added to every embedding, and the result passed to the Transformer encoder. During pretraining the embedded patch inputs I_i and the corresponding embedding output O_i are passed to individual Multilayer Perceptrons (MLPs), called [15, 70]. While the reconstruction head is trained to reconstruct the unmasked patch embedding for each masked position, using normal Mean Squared Error (MSE) loss. Therefore the authors use the term *joint* discrimination and reconstruction. After pretraining the SSAST is fine-tuned for a number of audio related tasks [15]. The results and other training details of the SSAST and AST will be described later in chapter 3.

Chapter 3

Related Work

In this chapter some of the work related to sound event detection and classification is presented. The chapter will limit the works presented to those directly related to sound event detection and classification in the marine audio domain, including both machine learning based methods and algorithmic methods. Some selected works from the terrestrial audio domain which employ machine learning based methods will also be introduced. As the machine learning models and architectures used in the terrestrial domain may well be applicable to the marine domain.

3.1 Marine Sound Event Detection and Classification

In this section the sound event detection and classification methods that have been used in the *marine* audio domain are presented. These will include methods utilizing various machine learning based techniques and some algorithmic methods. Note that though some of the related work presented as algorithmic methods in fact may utilize machine learning models, such as a simple feed-forward neural network. These are still described in this thesis as algorithmic methods due to the feature extraction techniques and simple network architectures employed by them, which differentiates these works from those employing more complex deep neural network architectures presented in this section.

3.1.1 Machine Learning Methods

This subsection will present some of the related work which primarily utilize machine learning based techniques within the marine audio domain. Firstly by introducing some of the work utilizing supervised techniques, followed by those utilizing variations of unsupervised techniques. A clear trend among the machine learning based approaches in the marine audio domain, is the use of CNN and RNN based supervised models, with some studies employing hybrid Convolutional Recurrent Neural Network (CRNN) architectures [12]. With far fewer studies seeming to utilize variations of unsupervised techniques. Furthermore, it is clear that such studies often focus on detecting sound events from one or a few *specific* species within the field of marine bioacoustics, rather than performing general anthropogenic and biophonic multi-label classification.

Supervised Methods

In [71] the authors use the ResNet18 [72] CNN architecture to perform binary classification of biophonic signals from orcas in hydrophone data. The authors train their model using data from the Orchive [73, 74], a dataset of approximately 19000 hours of underwater recordings collected using stationary hydrophones in British Columbia (Canada) over a 23 year period. The authors report that their best performing model achieve an Area Under the Receiver Operating Characteristic curve (AUROC) or 0.9523. The authors also evaluate the affect of model depth on the classification task, namely ResNet18, ResNet34, ResNet50 and ResNet101, and present that the deeper models only provide about a 1% performance increase on average on the, while requiring significantly more time for training and inference.

In [33] the authors propose a framework of classifying large amounts of hydrophone audio data collected using a network of moored hydrophones from Ocean Networks Canada (ONC)[1] to classify sounds from different marine mammals, including humpback whale, fin whale blue whale, sperm whale and white sided dolphins. The authors utilize a CNN-based model to perform feature extraction. Dimensionality reduction is then performed on the computed features using Principal Component Analysis (PCA), and the output fed to a Support Vector Machine (SVM) for classification. The authors report average accuracy of 94.48%, recall (sensitivity) of 83.26% and specificity of 97.36% on the multi-class classification task.

In [34] the authors evaluate five different deep neural network models on the task of detecting the vocalizations of North Atlantic right whales, with hydrophone data related to the Detection Classification Localization and Density Estimation (DCLDE) 2013 workshop [75]. The authors evaluate four different CNN-based architectures, LeNet [76], BirdNet [27], VGG [77], and ResNet [72], as well as a a Recurrent Neural Network (RNN)-based model which utilize one-dimensional convolutions [78]. With LeNet being the best performing model, with a recall of 0.946, which considerably beat the competition of the workshop, which achieved a maximum recall of 0.83. The DCLDE dataset the authors used was collected using a stationary hydrophone for a single week in 2009 in Massachusetts Bay, thereby severely limiting the spatial and temporal scales of the dataset. Meaning that the distribution and characteristics of the right whale calls might be significantly different for data recorded at other geographical locations or seasons. The authors therefore also evaluate the model ability to generalize to unseen data by first training on the DCLDE 2013 dataset, and evaluating the classifier on another dataset containing 33 days of recording from different locations (Georgia, North Caro-

[1]Ocean Networks Canada website: `https://www.oceannetworks.ca/`

lina, Virginia and Maryland), and compare its performance to a baseline detector trained on a separate Kaggle dataset [79]. With the deep learning based model significantly outperforming the baseline detector for all other location recordings. Thereby indicating that the deep networks are able to generalize to previously unseen data by learning the characteristics of the right whale calls [34].

In [80] the authors utilize a CNN-based model to perform joint detection and binary classification of orca calls hydrophone data from the OrcaLab[2] on Hanson Island (Northern Vancouver Island, Canada). The authors first utilize an algorithmic approach, inspired by [81], to remove any non-orca sound events from the dataset, and thereafter manually labeling the remaining sound events [80]. The labeled dataset was then used for training and evaluation by transfer learning of a CNN-based model from [82], originally trained for detecting vocalizations of birds in terrestrial environment. In [80] the authors report that the model achieved an AUROC of 0.89 [80], making it reasonably effective for the binary classification task.

Unsupervised Methods

In [83] the authors utilize the same CNN-based model and detected orca vocalizations as from [80], and perform unsupervised clustering on the detected vocalizations, with the goal to classify different *types* of orca vocalizations (pulsed calls), as these can vary between groups (pods) as "dialects" [84]. The pitch contour of the fundamental frequency of the detected vocalizations was computed, and a number of features computed from the resulting contour. Including minimum and maximum frequencies, change in frequency, velocity and acceleration of the pitch contour. The resulting features was then passed to the Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN) clustering algorithm, resulting in 13 detected clusters. Although the clusters did not necessarily clearly describe the known types of orca vocalizations, they show that such clustering is a promising method of automatically classifying orca calls [83].

In [85] the authors perform unsupervised classification of hydrophone audio data collected using fixed hydrophones off the coast of Chile, to detect the presence of blue whale calls in the recordings. The authors utilize both the Density-Based Spatial Clustering of Applications with Noise (DBSCAN)[86] algorithm and Gaussian Mixture Models (GMMs), with in total 46 different computed spectral and temporal features. Including Zero-Crossing Rates (ZCRs), spectral centroid, maximum frequency, Mel Frequency Cepstral Coefficients (MFCC) [87] and chromagram [88]. Their results show that the GMM using 8 expected clusters was able to separate blue whale calls from other sound events and outliers when verified by a bio-acoustic specialist [85].

In [31] a CNN-based model was applied to the task of multi-label classification of hydrophone data to detect and classify ship noise according to vessel category. With the authors reporting a classification accuracy of 79.2%.

[2]Orcalab website: `https://orcalab.org/`

3.1.2 Algorithmic Methods

In [21] the authors propose methods of performing detection and classification of specific biophonic and anthropogenic sound events, namely sperm whale Echolocation clicks and impulsive ship sounds, in hydrophone audio data. The hydrophone data used was collected from the NEutrino Mediterranean Observatory – Ocean Noise Detection Experiment (NEMO-ONDE) [89] submarine detector, primarily intended for the acoustic detection of high-energy neutrinos, which also contain sound events of anthropogenic and biological origin. The authors perform detection and classification of sound events as individual distinct tasks, firstly by detecting portions of the recordings in which a relevant sound event occurs, and thereafter performing feature extraction and classification of the detected segments.

The authors propose an algorithmic impulse detection method, performed on two separate frequency bands (1 kHz–5 kHz and 5 kHz–20 kHz). The method detects the onset and offsets of impulsive sounds by comparing the waveform magnitude envelope with the estimated local background noise. The magnitude envelope was computed as the running arithmetic mean of the waveform magnitude, and the background noise as the linearly interpolated running median of the waveform magnitude. Whenever the envelope rose above the background noise estimate by a factor of two, a section of the audio centered at the interval was marked as an impulsive sound events. For every detected sound event a total of nine spectral and temporal features (5 spectral and 4 temporal) was calculated for an interval of audio centered at the detected sound event. These included estimations of both spectral and temporal dispersion, asymmetry, concentration and Shannon entropy, and spectral location, for each detection event. These features was then passed to a feed-forward neural network with nine inputs, a single hidden layer of 25 hidden neurons using radial basis function units, and a single output neuron with logistic activation was used to model the probability that the detected event contained a sperm whale click $P(SWC)$ or an impulsive ship noise $P(ISN) = 1 - P(SWC)$. The authors present classification performance by Area Under the Receiver Operating Characteristic curve (AUROC), with values ranging between 0.74 and 0.98 depending upon number of hidden units used in the network and details of the testing dataset used.

This method was later used in [20] to detect impulsive signals. They also used short and long tonal signals from biophonic sounds such as dolphin whistles and baleen whale calls, and anthropogenic sounds from depth gauging devices called depth sounders, on data from NEMO-ONDE. Using a similar detection approach by detecting tonals from different target species in different frequency bands, using the <5 kHz band to detect baleen whale tonal calls, 2 kHz–20 kHz for dolphins, including orcas. Frequencies above 20 kHz was also used but had not detected any biophonic signals, but could be triggered by certain anthropogenic sound sources. After detection the same feature extraction and classification method was applied.

In [29] the authors propose a method to detect the low-frequency baleen whale calls from hydrophone recordings recorded in the southwestern Gulf of Maine. The method first performs spectrogram smoothing as described in [90], and thereafter tries to reduce the amount of long-duration tonal noise with little variation by subtracting an exponentially weighted running mean m from each frequency band of the smoothed spectrogram S, using equations 3.1, 3.2 and 3.3 [29].

$$A_{i,j} = S_{i,j} - m_{i-1,j} \tag{3.1}$$

$$m_{i,j} = (1-\varepsilon)m_{i-1,j} + \varepsilon S_{i,j} \tag{3.2}$$

$$\varepsilon = 1 - e^{\log(0.15)\frac{\Delta t}{T}} \tag{3.3}$$

Where Δt is the temporal resolution of the spectrogram and T a time constant set to 10 s in [29]. The authors explain that the value of T should be longer than the duration of the longest that is expected in the recordings. After which the authors perform broadband noise reduction by first inspecting successive frequency bands to detect where in the spectrograms these sounds occur. They then remove these noises by computing their duration and bandwidth, and setting the spectrogram values of these parts of the spectrogram to zero. Thereafter the authors estimate the pitch tracks of whale calls, perform feature extraction of these pitch tracks, and use these computed scalar features with Quadratic Discriminant Function Analysis (QDFA). The resulting model was then used to perform classification of the recorded hydrophone data.

In [91] utilized a variation of the spectrogram correlation algorithm described in [92], on marine acoustic data collected using a glider based recording platform. The algorithm utilizes two-dimensional kernels that are constructed to approximate the shape of certain whale tonal vocalizations. The kernels are cross-correlated over the spectrogram representation of the input audio, to produce a detection function that could be thresholded to produce detections. The detection method employed in [91] and [92] is that the latter constructed the kernel by synthetic linearly frequency modulated sweeps. Whereas in [91], the authors construct the kernel based on the amplitude contours of averaged examples of the target species vocalizations.

In [93] the authors utilize the detection and classification method as used in [20], including the algorithmic echolocation click and impulsive noise detection method from [21], for data collected using using a glider-based recording platform. Demonstrating that these detection methods *can* be used with glider based platforms.

3.2 Terrestrial Sound Event Detection and Classification

In this section some of the related work within the field of terrestrial sound event detection will be introduced. We will limit this section to only present the related work that utilize purely machine learning based techniques, and forgoing any algorithmic approaches, as these are the dominant method of performing sound event detection and classification [55], and because we are primarily interested in exploring the applicability of machine learning based techniques in the marine audio domain. We chose to introduce these related works for the terrestrial domain as some of these methods have successfully been adopted in the marine domain previously, as we have seen by the trend of applying CNN-based models for marine sound event detection and classification previously in this chapter.

Earlier studies in the terrestrial audio domain often utilized Gaussian Mixture Models (GMMs) or Hidden Markov Models (HMMs), often operating on features such as the Mel Frequency Cepstral Coefficients (MFCC) [94–98]. A common recent trend in sound event detection and classification in the terrestrial domain is, similarly to the marine domain, the use of CNNs, RNNs or hybrid CRNN based models that operate in the time-frequency domain[39, 55, 99]. Although it is most common for contemporary methods in the audio domain to operate on time-frequency representations of the underlying audio data, there are some studies which operate on the raw audio waveforms. Such as [100] in which the authors propose a generative deep neural network architecture that was applied to text-to-speech, creating naturally sounding voices.

Recently the transformer architecture has been more widely adopted in the several domains, with models such as the Vision Transformer (ViT) [36] being applied in the computer vision domain, and the Audio Spectrogram Transformer (AST) [14] in the audio domain, with promising results. As the AST and SSAST models will be evaluated for the marine audio domain in this thesis, we will describe these studies in greater detail in the following paragraphs.

In [14] the authors introduced the Audio Spectrogram Transformer (AST), based on the Vision Transformer (ViT) by [36] the AST adapted the transformer for the audio domain. In [14] the AST was evaluated on three different datasets; AudioSet [101], ESC-50 [102] and Speech Commands V2 [103]. All of which being audio classification datasets containing different types of sounds from terrestrial environments. The authors evaluate the model both with and without supervised pretraining on the image-classification dataset ImageNet [104]. The authors presents the results of two sets of experiments, the first providing the effects of ImageNet pretraining on model performance on just the AudioSet dataset. The second set of results present the model performance on ESC-50 and Speech Commands V2 with ImageNet pretraining only, and with ImageNet pretraining and AudioSet pretraining. An overview of the results from [14] can be found in table 3.1. Note that the authors of [14] present performance metrics for AST in a number of configurations for each task, but only the best performing results for each configuration is presented in table 3.1 here. Also note that the performance metric presented in [14], and in table 3.1, are not the same for every dataset. The authors of [14] likely use mean Average Precision (mAp) as the performance metric for AudioSet because this is the most common evaluation metric for AudioSet, although this is not explicitly described by the authors.

Task	Performance	Pretraining
ESC-50	88.7 ± 0.7 accuracy	ImageNet
Speech Commands V2	98.11 ± 0.05 accuracy	ImageNet
ESC-50	95.6 ± 0.4 accuracy	ImageNet, AudioSet
Speech Commands V2	97.88 ± 0.03 accuracy	ImageNet, AudioSet
AudioSet	0.459 ± 0.000 mAp	ImageNet

Table 3.1: Audio Spectrogram Transformer (AST) performance from tables 1 and 7 in [14] for a variety of audio classification tasks and in a variety of configurations.

When it was first introduced in April of 2021, the AST outperformed the state-of-the-art (SOTA) models on AudioSet [101], ESC-50 [102] and the SpeechCommands V2 [103] datasets [14]. While at the time of writing this thesis in July of 2022, the model has been superseded by at least three other models for AudioSet [105–107], three other models for ESC-50 [105, 108, 109] and one other model for SpeechCommands V2 [108], based on the ranking of model performance for these tasks on the Papers With Code website[3]. This can be a good example of that help showcase the rate at which progress is made within the machine learning audio domain today.

In [15] the authors evaluate the AST trained using a self-supervised framework, on a number of datasets, including AudioSet, ESC-50 and Speech Commands V2 as used in [14]. The authors call this version on the AST the Self-Supervised Audio Spectrogram Transformer (SSAST). The authors show that the SSAST achieve performance presented in table 3.2. Note that only the best performance for each dataset presented in [15] are shown in table 3.2.

Task	Performance	Pretraining
AudioSet-20K[4]	31.0mAp	ImageNet, AudioSet-2M
ESC-50	88.8 accuracy	ImageNet, AudioSet-2M
Speech Commands V2	98.1 accuracy	ImageNet, AudioSet-2M

Table 3.2: Self-Supervised Audio Spectrogram Transformer (SSAST) performance from table 1 in [15]

The authors compare the SSAST to a randomly initialized AST model without any supervised nor self-supervised pretraining, fine-tuned for audio classification on AudioSet. With the SSAST significantly outperforming the AST model, showing that the AST achieve mean Average Precision (mAp) of 0.148 while the SSAST achieves mAp of 0.31. However, the AST model pretrained in supervised fashion on AudioSet and/or ImageNet still outperforms the SSAST in the results presented in [15]. The results of the SSAST from [15] show that the self-supervised pretrain-

[3]Papers With Code website: `https://paperswithcode.com/`

ing can yield promising results if labeled data is limited, although the pretrained supervised techniques still outperform the SSAST.

3.3 Audio Datasets

There are several datasets used in the relevant literature relating to both the terrestrial and marine audio domain. In this section we will briefly mention some of the most notable audio datasets for both terrestrial and marine environments. An overview of some of these datasets can be found in table 3.3. In this project the hydrophone audio data from the GLIDER dataset is used, which will be introduced later in chapter 4.

Dataset	Target classes	Target examples
	Terrestrial	
AudioSet [101]	632	Sounds of: cat, dog, jet engine, thunder
ESC-50		
SpeechCommands V2 [103]	35	Utterances of: "backward", "cat", "follow", "visual"
DCASE 2019 (AudioSet subset) [28]	10	speech, dog, cat, vacuum cleaner
DCASE 2013 [110]	16	cough, drawer, keyboard, phone
	Marine	
DCLDE Oahu [37]	17	Sperm whale, minke whale, bottlenose dolphin, melon-headed whale
The Orchive [73]	1	Orca vocalizations and clicks
Watkins Marine Mammal Sound Database [111]	>60	Bowhead whale, fin whale, humpback whale, bottlenose dolphin, snapping shrimp
NOAA PIFSC dataset [112, 113]	12	Anthropogenic sound, minke whale, fin whale, noise from the recording equipment

Table 3.3: Examples of some notable terrestrial and marine audio datasets.

Chapter 4

Method

In this chapter the methods utilized in this project are outlined and described. This will include details regarding the data processing pipeline that is utilized to enable deep learning based models to be applied to the GLIDER dataset. As well as details such as the metrics we will utilize to evaluate and compare the models' performance, tools used to implement the data pipeline, and details regarding training, validation and testing. The chapter will first describe relevant details of the model implementations this project utilizes.

4.1 Models

This thesis uses the implementations of the Audio Spectrogram Transformer (AST) from [14] and Self-Supervised Audio Spectrogram Transformer (SSAST) from [15], with only minor alterations to accommodate for differences in available hardware in computing environment used during training. This is done with the intent to keep the implementation as close as possible to the original implementation of the models, such that comparison of the model performance in the marine audio domain be as fair as practically possible to the original terrestrial domain the models were trained and evaluated for. We use the ResNet18 implementation used in our preliminary work in [1], as it was introduced and described in [72]. This is done to make the performance of the ResNet18 applied to the GLIDER dataset in this project as comparable as practically possible to the results from our preliminary work using the DCLDE Oahu dataset [37].

4.2 Dataset

The dataset used in this project, as mentioned in chapter 1, is the hydrophone audio data collected using an underwater glider in the GLIDER project conducted by Akvaplan Niva in 2018. During this project a glider was deployed on the 15th of March and was recovered on September the 1st, 2018. A pre-programmed survey track was designed to cover both the continental shelf and the shelf break.

The glider was equipped with a JASCO AMAR G4 hydrophone (JASCO Applied Sciences Ltd.) to detect and record cetacean vocalizations and other underwater noise, operating continuously during periods of 10 minutes during descent to 200 m depth. This was done in order to avoid the pump motor noise on ascent and preserve battery power. The hydrophone recording unit had a sampling frequency of 128 kHz and was mounted in the glider's aft wet space close to the buoyancy bladder, with the hydrophone on top of the glider just in front of the rudder. Due to the large amount of data collected, a subset of the data of ecological interested was selected for this study.

The GLIDER dataset used in this project, consists of 5249 single channel hydrophone audio recordings in uncompressed .wav format, recorded using a sampling rate of 128 kHz, yielding a Nyquist frequency of 64 kHz. The total size of the dataset is 1019 GB. The starting timestamp of the recordings are determined from their filenames, an example of the filenames and sizes can be found in table 4.1. The expected duration for each file is 10 min, but 1096 of the files are shorter than this. These files might be shorter due to an error occurring while recording, which caused the recording to end prematurely. These shorter files are all used alongside the full length files for this project. Any files that are otherwise corrupted and not possible to read using typical Python [53] audio tools such as librosa [52] are discarded in this project.

Size	Filename
220MB	`pa0298bu_003_180731_235322.wav`
220MB	`pa0298bu_002_180731_234323.wav`
220MB	`pa0298bu_001_180731_233319.wav`
132MB	`pa0298au_005_180731_221025.wav`
220MB	`pa0298au_004_180731_220026.wav`
220MB	`pa0298au_003_180731_215027.wav`
220MB	`pa0298au_002_180731_214028.wav`
220MB	`pa0298au_001_180731_213025.wav`
128MB	`pa0297bu_005_180731_210241.wav`

Table 4.1: Example filesize and filename of some files from the GLIDER dataset. The starting timestamp is determined by the last 12 digits of the filename, describing timestamp information in the format `YYmmDD_HHMMSS` (Y = year, m = month, D = day, H = hour, M = minute, S = second)

Only considering the short duration files, the short files have a mean duration of 330.602 s with a standard deviation of 158.520 s, with a median duration of 334.879 s, and minimum and maximum duration of 0.478 s and 599.472 s respectively. Considering all of the recordings of the dataset, the recordings have a mean duration of 543.749 s with a standard deviation of 131.282 s, a median duration of 600 s, with minimum and maximum duration of 0.478 s and 600 s respectively. The file duration distribution of the GLIDER dataset may be more intuitively understood by the histogram found in figure 4.1.

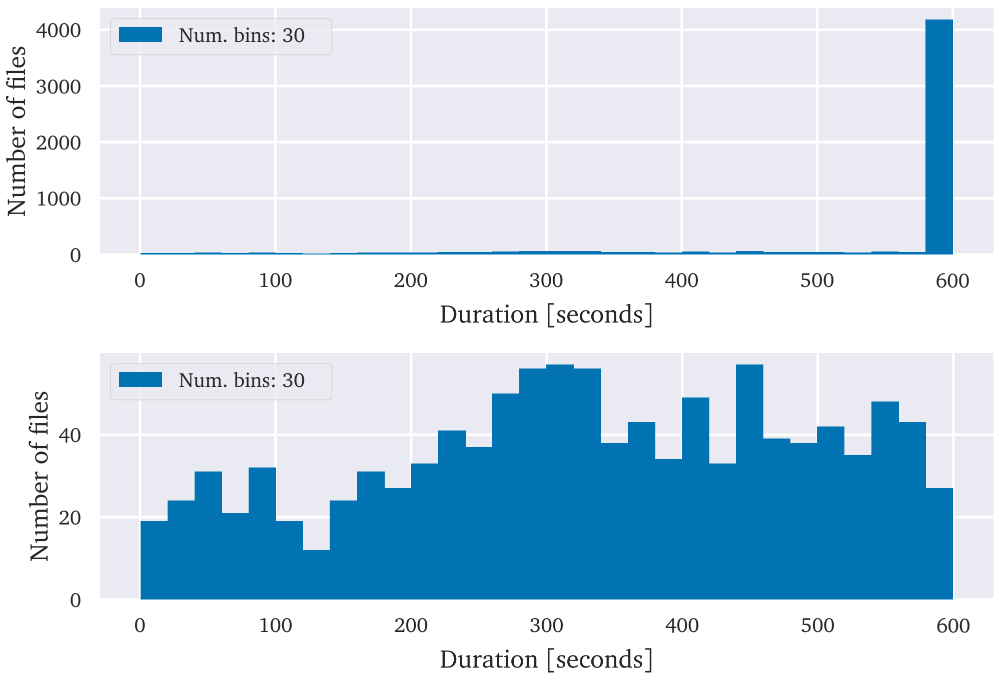

Figure 4.1: Histogram showing the file duration distribution in seconds. Top plot shows the distribution of file duration in seconds for all uncorrupted files in the GLIDER dataset. The bottom plot shows the distribution for files with duration less than the expected 10 min

The recordings of the GLIDER dataset used in this project was recorded over a 6 month period in 2018, with the first recording made on the 20th of April 2018, and the last recording made on the 8th of September 2018. Although spanning approximately 6 months, no recordings of the dataset was made in May. Figure 4.2 show the number of hours of recorded audio by day in the GLIDER dataset. From this we can tell that recordings was made for relatively short durations in April, June and September, and that the majority of recordings in the dataset was made in July and August. The total duration of all uncorrupted audio within the dataset is 792 hours, 38 minutes and 59.774 seconds.

The distribution of recorded audio by day, as presented in figure 4.2, might be significant in relation to the task of sound event detection and classification. If there are some seasonally dependent sound events, these may be over- or underrepresented in the dataset. Considering that some marine species change their behaviour and vocalization according to season, such skewed representation of such sound sources in the GLIDER dataset is expected. An example of this is the song of the humpback whale, which increase in male individuals during the mating season [114]. This may indeed also be the case for both anthropogenic sound

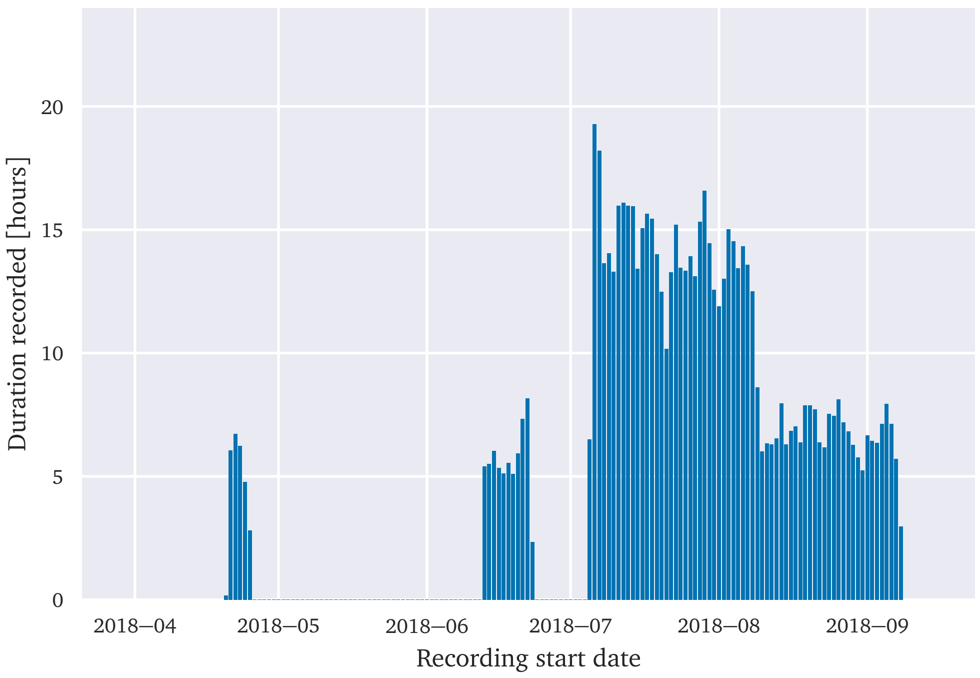

Figure 4.2: Hours of audio recorded by day in the GLIDER dataset used in this project

events, such as noise from geological survey equipment such as seismic airguns [115], and for other ambient sound events such as iceberg cracking, rain, lightning and geological sound sources. Furthermore, there might exist sound sources that do not occur annually, which our model should be subjected to during training. However, as this is an issue with the original data collection, this is not considered in greater detail in this project. Although it should be mentioned that this may be detrimental to the performance of typical machine learning models, trained on such skewed data. If such models are used as-is on new data where the distribution of sound event classes is significantly different to the data used during training, the models may perform poorly compared to results from testing on held-out data from the original dataset. However, as we are unable to effectively and practically measure such a disparity in testing and deployment results, no further investigation of this issue will be presented in this thesis.

GLIDER Dataset Labels

The labels for the GLIDER dataset provided by Akvaplan Niva, originally had the form of two distinct .xslx spreadsheet files, one for biological sound sources, and another for anthropogenic sound sources. After the data from each file was aggregated, the resulting list of labels take the form of a list of starting and ending timestamps associated with one or more sound sources. An example of this aggregated list of labels can be found in table 4.2. These indicate the sound source of the sound event located by the starting and ending timestamps, and are either considered as "Biotic" or "Anthropogenic" labels for our purposes. These sound sources are for example, ship noise, airguns, fin whales or pilot whales.

Start Time	End Time	Label	Sound Source
2018-07-30 16:26:00	16:59:00	Biotic	Sperm whale
2018-07-30 16:45:00	16:46:00	Biotic	Unid. Delphinid
2018-07-30 17:49:00	18:05:00	Biotic	Sperm Whale
2018-07-30 17:51:00	17:53:00	Biotic	Blackfish
2018-07-31 01:21:00	01:30:00	Anthropogenic	Shipping
2018-08-03 17:00:00	17:30:00	Anthropogenic	Aiguns
2018-08-29 20:16:00	21:04:00	Biotic	Fin whale

Table 4.2: Example of parsed and aggregated labels of the GLIDER dataset. The date information of the ending timestamp is omitted from this table to improve readability, as the starting and ending date is the same for all of these examples.

There are 546 individual labels in the dataset, each marking a specific starting and ending timestamp alongside a biophonic or anthropogenic sound source. The distribution of the duration of the labels can be found in figure 4.3.

An important detail we should be aware of is that the list of labels do not contain any negative instances, due to their use for ecological studies. Meaning that the labels only indicate that some biophonic and/or anthropogenic sound event occurs in the interval defined by the label starting and ending timestamps, but no label indicate that an interval contains no such sound events. But our classifiers need to be subjected to both positive and negative instances during training, such that the models are able to distinguish positive instances from negative instances when presented with new unseen data. Therefore, in this project, we consider any file that does not overlap temporally with any label as a negative instance. This is an important point when considering the results of our multi-label classifiers later in this thesis. As only a small portion of the dataset has been labeled, considering any file with no overlapping labels as having a negative example will very likely lead us to label files that do actually contain some anthropogenic and/or biophonic sound event as negative instances. Thus indicating to our model during training that the example does not contain anthropogenic nor biophonic sounds, while the input actually *does* contain such sounds. If the number of such instances is high relative to the overall dataset, we might expect that our model would not be able to converge to some solution, because no real pattern would exist between the input and the desired target that our model could learn. For this reason, one good indication of high noise in the dataset labels, either from such false negative instances, or from errors within the original labels, can be lack of model convergence during training.

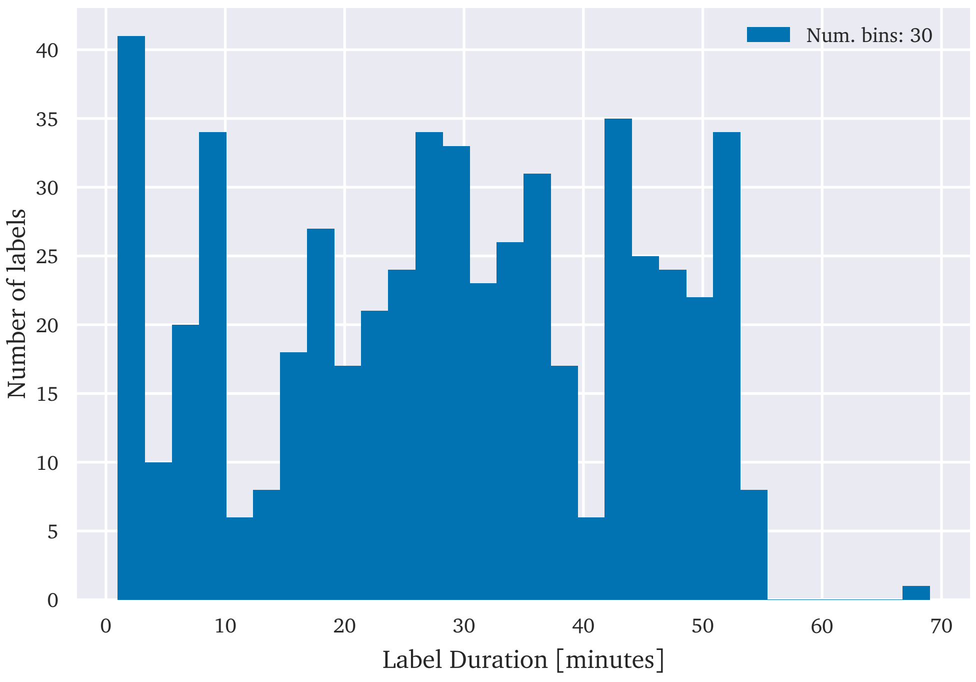

Figure 4.3: Histogram showing the distribution of label durations of the GLIDER dataset, calculated as the difference between the label ending and starting timestamps, displayed in seconds.

4.2.1 Preprocessing

In this section the preprocessing steps applied to the GLIDER dataset in this project are described. This includes data cleaning, transformations and normalization techniques applied to the original recordings of the dataset, enabling us to use deep learning based models to perform multi-label classification of the audio data.

File Timestamps

The filename of each audio file in the GLIDER dataset contains metadata that is used to determine the starting timestamp of the recording, as well as other metadata which is not used for this project. We need to be able to determine both the starting and ending timestamps of every recording in the dataset, such that we may find the labels that overlap temporally with the recording. As the filename of the recordings only provide us with the starting timestamp of the recordings we need to determine the duration of each file to infer the ending timestamp of the recording. In this regard we have two options, the first being to read the entire file contents into memory and calculate the duration D of the recording in seconds using the number of samples in memory N and the sampling rate S_r, expressed in hertz, of the recording as $D = \frac{N}{S_r}$. Another more memory-efficient option is to read only the parts of the binary header information from each file that describes the number of samples, and sampling rate, of the recording. This option is more memory-efficient because in this case we only need to read 10 bytes in total from each file, rather than several hundred megabytes per file. However, as we know some of the files have unexpected duration, possibly due to errors when recording, the header information for some of these files might be incorrect. Such that the number of samples stated in the file header differs from the number of samples actually contained in the file. If we then use the number of samples and sample rate stated in the file header to compute the duration, and thus compute the ending timestamp using this information, our result would be incorrect. Later when we find the temporally overlapping labels to the file, our incorrect ending timestamp could cause us to set incorrect labels for the file. In turn, should a significant number of files be corrupted this way, this could lead to reduced model performance due to increased noise introduced to the labels by our incorrect labeling. For this reason we decided to iterate through the files of the dataset, read their entire contents into memory, and compute the duration and ending timestamp of every file by the number of samples actually found within the file. The result of this computation is then stored on the local filesystem, and only re-computed if the audio files found within the dataset directory changes. In which case the new result would be stored alongside the old result in a cache directory.

This method of computing the ending timestamps of the files enables us the benefit of both approaches. Firstly we are able to check the starting and ending timestamps of every recording when we initialize our dataset, without the need to store the clip in memory every time. By doing so we can implement our data pipeline in such a way that it only reads the file contents into memory when required, when the input batch is created by the PyTorch [116] dataloader. This reduces the memory overhead during initialization, and makes local development easier by reducing heavy I/O workloads required every time the dataset is initialized. Secondly, we reduce the risk of applying incorrect labels to the recordings due to incorrect ending timestamp of the recordings, as the duration of the recordings is inferred using the actual number of samples stored in the file and the sample rate of the recording rather than the possibly faulty header information.

Clipping

In the original implementation of the AST in [14], the authors adapted the ViT from the vision domain by [36] to the audio domain. The authors provide the implementation of their adapted model, alongside pretrained model weights, through a git repository[1]. In their implementation they enable instantiating the AST from both the original ViT parameters trained on ImageNet [104], and their own para-

[1]GitHub repository with implementation of [14]: `https://github.com/YuanGongND/ast`

meters fine-tuned on AudioSet [101]. We utilize the AudioSet pretrained AST model parameters in this project, as these are trained for audio data specifically. The implementation for the AudioSet fine-tuned AST version requires that the input time-dimension of the input spectrograms be 1024 elements, and the number of input mel-bands to be 128. This leads to some limitations regarding our choice for number of Fourier transforms windows and number of filters in our mel-filter bank that we use to compute the Mel-spectrogram for a given input. Furthermore, as AudioSet is made up of 10-second audio clips, the AudioSet fine-tuned version of the AST consequently is trained for such 10-second clips. Therefore, to keep the details of our training and testing pipeline as similar as meaningfully possible to the original AST implementation, and to ensure that the details for the AST and the SSAST used in this work as comparable as possible, we clip the recordings of the GLIDER dataset into 10-seconds clips. This clipping also acts as a data augmentation technique, increasing the number of examples available to us. We also overlap each clip with the previous clip by 4 seconds with the intention to try to ensure that those transient events that are split into separate clips, have a higher likelihood to be represented in their entirety in the next overlapping clip.

Clipping in this manner transforms our dataset consisting of 5249 (mostly) 600 s audio recordings into 469113 10-second audio clips. The distribution of these clips by their class labels can be found in figure 4.4. Note that because the class labels are independent, meaning that anthropogenic and biophonic sound events may overlap temporally, the optimization task is a multi-label classification task specifically. Therefore, the "Both" column in figure 4.4 represent clips that overlap temporally with both anthropogenic and biophonic labels, whereas the "Neither" column represent clips that *do not* overlap temporally with *any* labels.

Our implementation of the clipping functionality is such that we do not require the actual audio data loaded into memory to perform the clipping. Clipping is done using only the list of audio files found in the local filesystem directory, their starting and ending timestamps, their known sampling rate, and the desired clipping parameters, such as the clip duration and overlap in seconds. This information is used to calculate the index of the first and last samples of the clip relative to its underlying audio file. When the clip data is required for some downstream computation, such as computing the mel-spectrogram, only the samples between the starting and ending indices from the underlying audio file is read into memory. By implementing the clipping functionality this way, we ensure that the underlying audio data stored on the local filesystem is never actually altered. This ensures that our preprocessing pipeline be adaptable for future work using the GLIDER dataset, as using the same clipping parameters, if any, might not be applicable for future projects. After this clipping is applied, we iterate over the clips and try to read them into memory to verify that none of the clips are corrupted or otherwise cause some error during training. This is done only once before the clip information (filename, starting and ending sample indices) are stored to the filesystem, and is only ever re-computed if the clipping parameters or the list of audio files change.

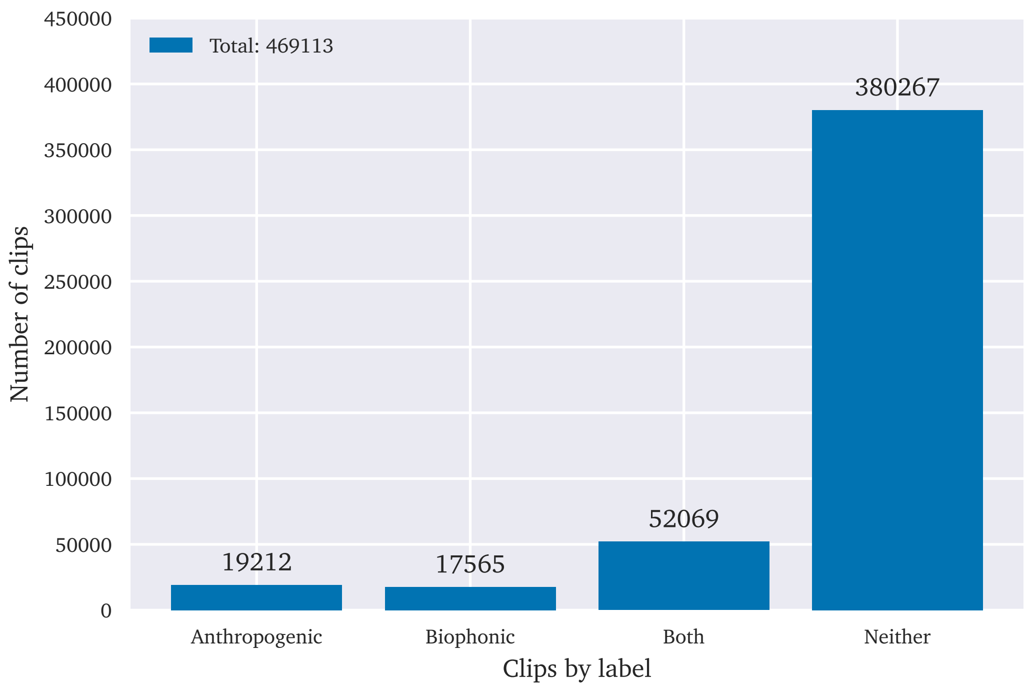

Figure 4.4: Number of clips by class labels

As previously mentioned the GLIDER dataset consists of audio recordings sampled at 128 kHz. This sampling rate is significantly higher than that typically used for terrestrial audio. The recordings of ESC50 for example, typically use a sampling rate of 44.1 kHz [102]. Using audio data with lower sampling rate lowers the maximum frequency (Nyquist frequency) we are able to effectively record. In terrestrial environments, this might indeed not be an issue to the performance on the final prediction task. However, due to the nature of acoustic signals in aquatic environments, using lower sampling rates could be detrimental to the model performance due to loss of information from the higher frequency bands. Considering that some soniferous marine animals utilize far higher frequency ranges than those used in human auditory perception, we risk loosing these high-frequency calls if we lower the sampling rate. To clarify, even though the raw audio data of the GLIDER dataset is already recorded at 128 kHz, we could downsample the recorded signal in memory. In doing so we would effectively be compressing the input signal used to create our input mel-spectrograms. For this reason, we chose to use the original sampling rate of the recorded audio of the GLIDER dataset, not performing any downsampling. Meaning that for this project, we use a sampling rate of 128 kHz, compared to the sampling rate of used in the original AST paper of 16 kHz [14, 117]. This disparity in sampling rate used in this project and that of [14] yields differing Nyquist limits, of 64 kHz and 8 kHz respectively. However, because we need to ensure that the dimensions of our mel-spectrograms are equal to those used by [14], we still need to utilize the same number of mel-filters, being 128. This means that the number of frequencies spanned by each mel-filter used in this project will be greater than that for [14]. This lowers the frequency resolution of our spectrograms compared to that of [14], which might be detrimental to the performance of the model for the final prediction task in this project.

4.2.2 Normalization

In many cases machine learning based models, and especially deep learning based models, perform better if the distribution of their input values have zero mean and unit variance. Therefore, after clipping, we iterate over all the clips and compute the mean and standard deviation of the mel-spectrograms. Such that we may scale the spectrograms accordingly. This is done once for any set of clipping and spectrogram parameters. The spectrogram parameters include the length of the Fourier transform windows, as well as the hop length between successive windows, and the number of mel-filters to use to convert the frequency spectrogram to mel-spectrogram. If these values, or the list of audio files stored on the local filesystem change, the mean and standard deviation of the spectrograms must be recomputed.

Using a clip duration of 10 seconds and clip overlap of 4 seconds, Fourier transform window length of 3200 with hop length of 1280 and 128 mel filters, we first compute the mel-scaled spectrogram. In figure 4.5 we can see the distribution of spectrogram values as a histogram, computed from 500 randomly selected clips.

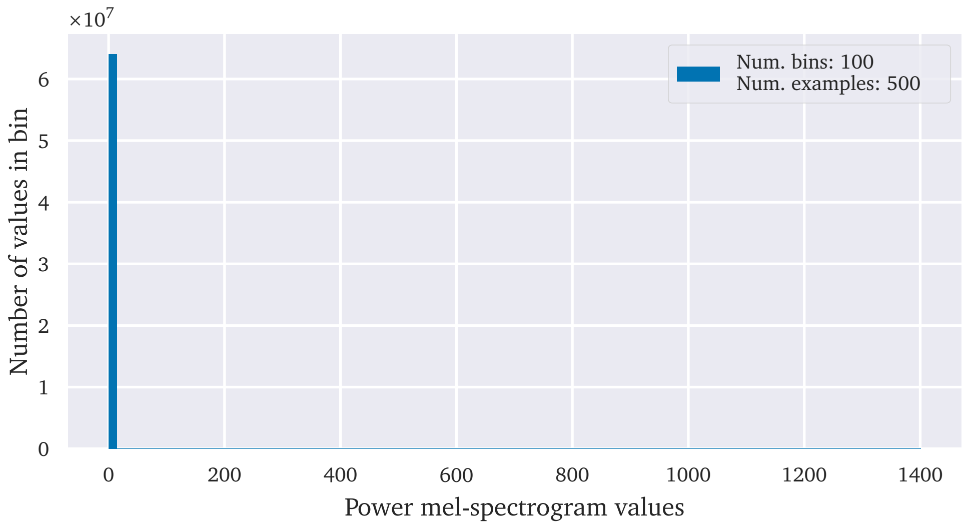

Figure 4.5: Distribution of power scaled mel-spectrograms values before normalization

In figure 4.5 using a linear y-axis, it seems that there are only values falling within the first column. However, if we use a logarithmic y-axis for the plot, as found in figure 4.6, it becomes clear that the power mel-spectrograms indeed does have values falling into other columns although comparatively few in relation to those falling in the first column.

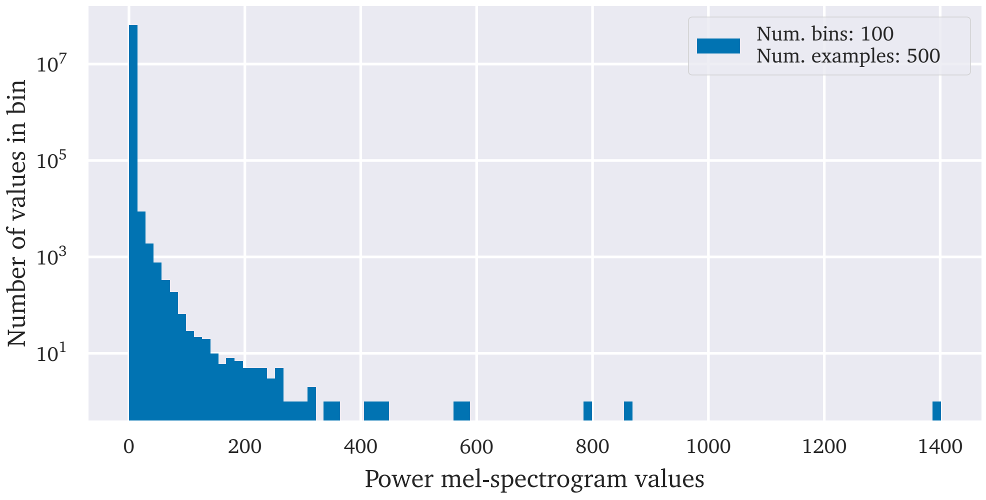

Figure 4.6: Distribution of power mel-spectrograms values before normalization, using logarithmic y-axis

One method of converting this distribution to one more similar to a normal/Gaussian distribution, is to compute the logarithm of the spectrogram values. However, as described in chapter 2, the decibel scale which can be used to describe the sound level is logarithmic. Therefore, if we convert the power mel-spectrograms to the sound intensity level using the decibel scale, the distribution of spectrogram values will be more normal than by using the power mel-spectrogram. Using the same random clips as used in figures 4.5 and 4.6, we plot the distribution of mel-spectrogram values using the sound intensity level in the decibel scale in figure 4.7.

From figure 4.7 we see that the distribution of spectrogram values is more similar to the normal distribution compared to the original value distribution from figure 4.5. However, it is clear that the mean value and the standard deviation of the distribution is still not optimal, as the distribution does not have zero mean and unit variance. Therefore, we compute the mean m and standard deviation std of these decibel scaled mel-spectrogram values, and use them to normalize the input decibel mel-spectrograms S using equation 4.1. The normalization is done by subtracting the mean value from every element of the spectrogram, and dividing the result by the standard deviation. This ensures that the distribution have zero mean and unit variance. The distribution of decibel scale mel-spectrogram values after this normalization can be found in figure 4.8. Where we now see that the distribution is centered at x=0 and has a variance closer to 1.

$$Normalized = \frac{S - mean}{std} \tag{4.1}$$

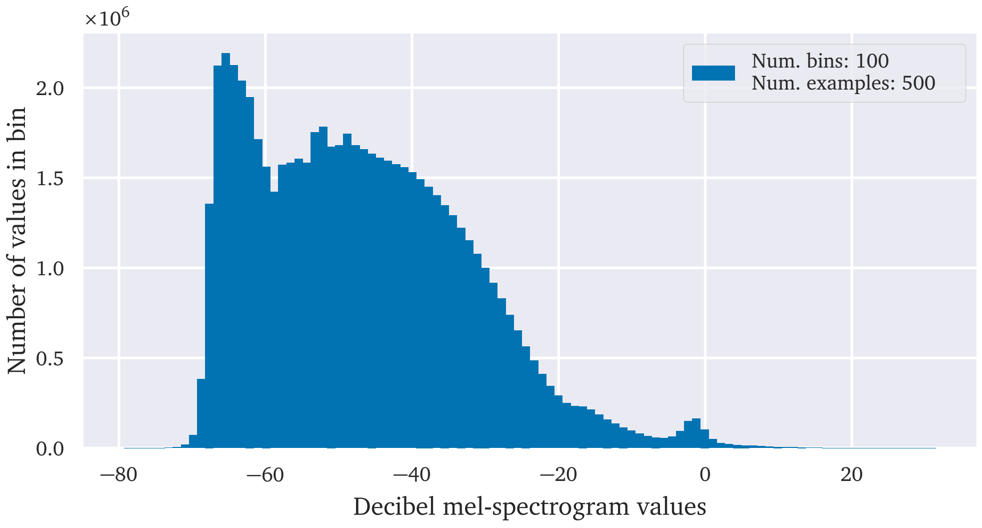

Figure 4.7: Distribution of decibel scale mel-spectrograms values before normalization

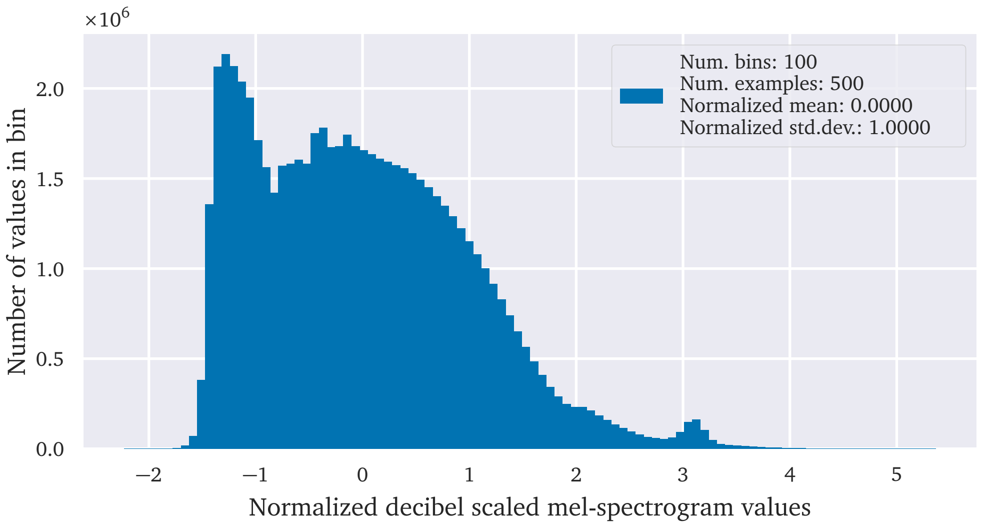

Figure 4.8: Distribution of decibel scale mel-spectrograms values after normalization

4.2.3 Data Augmentation

To increase the number of clips available during training, we apply a data augmentation technique called *SpecAugment* [118]. SpecAugment operates on time-frequency representations of audio data, such as spectrograms or mel-scale spectrograms, and performs three main operations. SpecAugment performs random masking of a predefined number of adjacent frequency bands, random masking of a predefined number of time-frames, and stretches the spectrograms along the frequency dimension [118]. The masked values are replaced with zero, this being the mean value after normalization. Some examples of spectrograms augmented with this technique can be found in figure 4.9. In this project we only apply SpecAugment for the training sets of the supervised techniques, and not for the SSAST. This is because the SSAST implementation performs random spectral masking during training by default, referred to as Masked Spectrogram Patch Modeling (MSPM) [15], such that applying SpecAugment would interfere with the SSAST optimization task. For any original spectrogram, we generate three new ones by SpecAugment, and use all four during training. We do not apply SpecAugment for the validation nor test sets. For the time and frequency masking, we use the default implementations available with Torchaudio[2] [119]. For time-warping we use the implementation from [120].

4.2.4 Dataset Split

To train and test the trained models, we split the clipped dataset into training, validation and testing subsets. During training, for every epoch, we perform perform a validation loop, making predictions on the validation set and computing the loss for the validation set. We then compare the training loss and the validation loss, such that we may perform early stopping to ensure that the model does not overfit to the training data. Early stopping is applied if no improvement is found in the validation loss after 3 consecutive epochs. Note that no backward pass is performed during the validation loop. Once training is complete, we test the model using the testing subset, such that we can evaluate the model performance on data the model has never seen during training nor validation.

However, because we are using both a supervised and self-supervised model in this work, we must split the data differently for each model. For the supervised model we use 64% of the clipped data for training, 16% for validation and the remaining 20% for testing. From section 4.2.1 we know that the distribution of labels is unbalanced, and that the majority of the dataset consists of have negative clips, meaning clips that do not have any anthropogenic nor biophonic labels. This *can* cause overfitting issues if the training data maintain the same imbalanced distribution, as the model may learn to merely predict that the input is negative regardless of the input data. Thus achieving relatively good accuracy during training, but greatly reduced performance on the testing set. Therefore we try to ensure

[2]Torchaudio GitHub repository: `https://github.com/pytorch/audio`

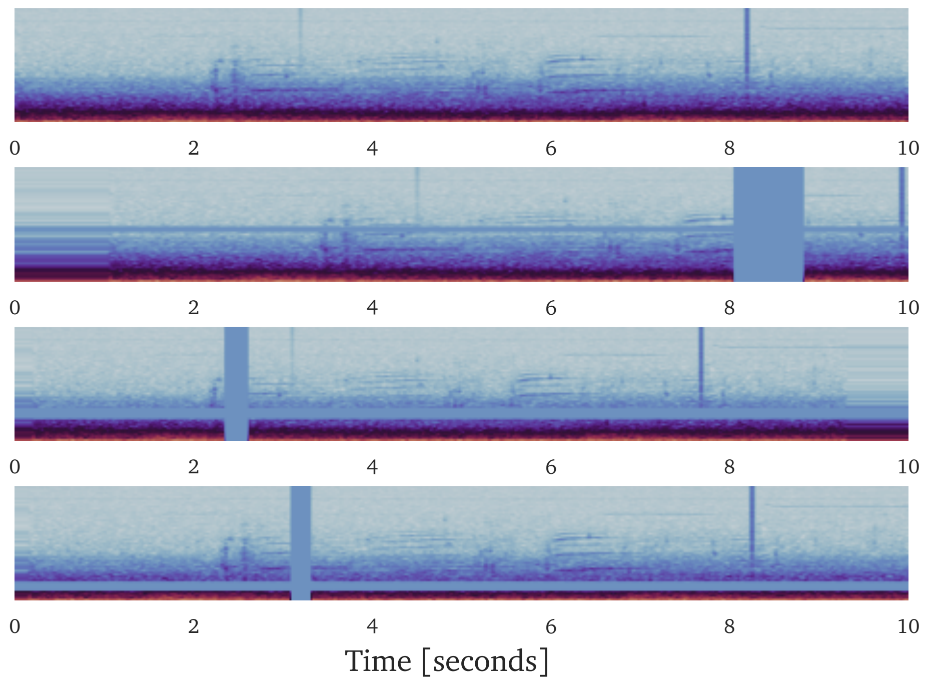

Figure 4.9: Examle of SpecAugment [118]. The top plot shows the original spectrogram, while the tree below shows how SpecAugment masks and warps the original spectrogram.

that the model actually attempts to learn the relationship between inputs and target values by balancing the training set according to target labels, while ensuring that the testing data maintain the same or similar distribution to the original label distribution among the clips.

For the supervised framework, we use 80% of the clips for training and the remaining 20% testing. We then use 20% of the training data for the validation loop. Resulting in 64% of the clips being used for training, 16% used for the validation loop, and 20% for the testing data. An example of the train, validation and testing subset splits can be found in figure 4.10. We perform data augmentation on only the training data.

Train 64%
Val 16%
Test 20%

Figure 4.10: Train, validation and testing dataset splits

For the self-supervised framework, we need to split the dataset into pretraining and fine-tuning subsets, and split each of these subsets into their own train, validation and testing subsets. We use 80% of the entire clipped dataset for self-supervised pretraining, and the remaining 20% for the supervised fine-tuning.

For the pretraining subset we do not *technically* need to perform any testing loop, but we have chosen to perform train, validation and testing loops for the pretraining task such that we may inspect the performance of the model during this part of the training process. As this allows us to better understand if the model is overfitting during the pretraining task, by examining the differences between validation and testing performance during this phase of the training process.

For the train, validation and testing splits we follow the same splits for both the supervised and self-supervised frameworks, using 64%, 16% and 20% of the data pool for each split respectively. The only differences being that in the self-supervised framework the samples for fine-tuning are drawn from a 20% subset of the clips, because the other 80% of the clips are used during self-supervised pretraining. An example of the pretraining and fine-tuning splits for the self-supervised framework can be found in figure 4.11. During the pretraining task, we do not perform any augmentation, because the implementation of the SSAST by [15] performs masking of the input spectrograms by default. Such that if we perform any masking, as performed by SpecAugment, we would interfere with the masking performed by the SSAST implementation. For the fine-tuning task however, we do perform augmentation in the same manner as for the training subset of the supervised framework.

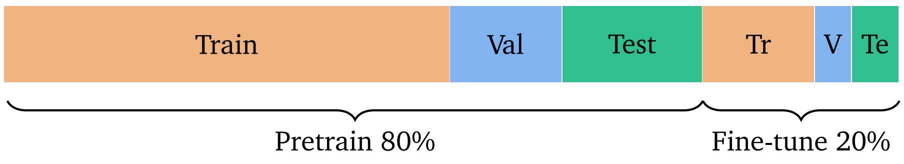

Figure 4.11: Pretraining and fine-tuning subsets with their corresponding train, validation and test subsets used with the self-supervised model. 64% is reserved for training, 16% for validation and 20% for testing, from both the pretraining and fine-tuning subsets.

4.3 Metrics

For all models used in this project, we compute the same set of performance metrics for each model during validation and testing, for the final task of multi-label classification to detect and classify anthropogenic and biophonic sound events. For the pretext training task of the self-supervised framework, we do not compute these metrics, as these would not be comparable to the metrics for the supervised framework due to the differing optimization task and dataset split for the pretraining and fine-tuning tasks. Because the distribution of label combinations are heavily imbalanced, we compute the support-weighted version for each metric, such that the model is penalized for constantly predicting the majority class, which can occur with imbalanced datasets. Meaning that we compute the metric for each class of the predictions and target values separately, and then compute the weighted average of each metric value using the class support as the weight. The class support is the number of instances of the class in the testing dataset.

The performance metrics used in this thesis is computed using the model prediction and target matrices. The predictions take the form of a 2-dimensional matrix of floating point values in the range 0–1. The target take the form of a matrix of the same dimensions, but where the values are binary integers, meaning they are either 0 or 1. Each row of the prediction matrix represent the model prediction of a single input example. Similarly, each row of the target matrix represent the target label values for that input example. The label order of the prediction and target columns is kept alphabetical according to the label name. Such that the first column of the target matrix describe whether the input example contain an anthropogenic sound event, and the second column describe whether the example contains a biophonic sound event. Similarly, the first column of the prediction matrix can be interpreted as the model confidence that the input contains an anthropogenic sound event, or its confidence that the input contains a biophonic sound event for the second column of the prediction matrix. Examples of the prediction and target matrices, $\hat{y}$ and y respectively, can be found in figure 4.12.

$$\hat{y} = \begin{array}{cc} \text{Anth.} & \text{Bio.} \\ \left[\begin{array}{c} 0.01 \\ 0.81 \\ \vdots \\ 0.23 \end{array}\right. & \left.\begin{array}{c} 0.2 \\ 0.42 \\ \vdots \\ 0.97 \end{array}\right] \end{array} \text{N} \qquad y = \begin{array}{cc} \text{Anth.} & \text{Bio.} \\ \left[\begin{array}{c} 0 \\ 1 \\ \vdots \\ 1 \end{array}\right. & \left.\begin{array}{c} 0 \\ 0 \\ \vdots \\ 1 \end{array}\right] \end{array} \text{N}$$

Figure 4.12: Example prediction $\hat{y}$ and target y matrices.

In this thesis we present the following support-weighted performance metrics for all presented models: accuracy, precision, recall, F1 score, Area Under the Receiver Operating Characteristic curve (AUROC) and mean Average Precision (mAp), as these are common in machine learning in the audio domain and for classification tasks in general and can give us a good understanding of how well the models perform. We compute these metrics using their standard implementation in the TorchMetrics[3] Python library [121].

4.4 Tools

To perform the training, validation and testing loops, we use the PyTorch Lightning (PL) python library [122]. PL enables us to implement models and datasets, including the training, validation and testing loops in a hardware agnostic manner. This has the advantage that the same pipeline is not hardware dependent, making

[3]TorchMetrics website: `https://torchmetrics.readthedocs.io/en/latest/`

local development simpler, while also enabling us to easier adapt the pipeline to the available computing resources. Furthermore, by using PyTorch Lightning (PL) much of the implementation details that can cause issues with the pipeline, which in turn may cause incorrect results, unexpected bugs or crashes, are abstracted away. Such that the chance of introducing these issues to the training, validation and testing pipelines can be reduced. We expected that the computational requirements necessary to train the AST and SSAST would be sufficiently great that the pipeline would need be distributed among several Graphics Processing Units (GPUs), and possibly several computing nodes (machines), because the authors of the original AST and SSAST papers state that their pipeline utilize 4 GPUs [14, 15, 117]. We therefore decided to use the Idun computing cluster [123] provided by NTNU for training, validation and testing.

The Idun computing cluster is a SLURM managed High Performance Computing (HPC) cluster, where we can define and schedule compute jobs, specifying the required hardware for our job(s). During training, validation and testing, for all models used in this work, we use a single compute node, using 4 GPUs, and a minimum of 24 CPUs. We chose to run the pipeline on a single compute node, because all models used in this work fits easily within a single GPU, such that using distributed training on multiple nodes would not provide any reduced training time or other benefit. But rather act as a bottleneck when the compute nodes perform synchronization of model gradients after the backward pass. The number of GPUs used during training is subject to the same type of limitation, where adding more GPUs for training does not necessarily always improve training time due the synchronization of tensors from one GPU to another. Therefore the optimal number of GPUs will likely vary between tasks, models and datasets. We chose to use 4 GPUs through a series of informal test, as this seemed to be a good trade-off between increased training time and time spent in the job queue on the cluster. If we increased the number of GPUs used, the job would often spend a long time in queue before being allocated the required hardware resources. The type of GPU used for each training run performed, in this work, depends on which GPUs were available when our job was allocated on the cluster. This was done such that we could further reduce the time our job would spend waiting to be allocated, due to high demand for the most powerful GPUs on the cluster. The pipeline has therefore been run several times, on three types of GPUs: Tesla P100-PCIE-16GB, Tesla V100-PCIE-32GB and NVIDIA A100-PCIE-40GB.

The decision to use PL however, while making the development process of the training, validation and testing pipeline simpler, requires that the PyTorch models used not assign any tensors to explicit devices. In the original implementation of the SSAST however, this is indeed the case. Therefore we forked[4] the original SSAST implementation[5], and made the necessary alterations to ensure that the SSAST model not inject any such device dependencies into the training pipeline.

To log and track model performance metrics, to monitor system performance

[4]Forked SSAST GitHub repository: `https://github.com/MartinMoan/ssast`

[5]SSAST GitHub Repository: `https://github.com/YuanGongND/ssast`

during training, to verify the model input, and to track changes to model performance across runs, we used the Machine Learning Operations (MLOps) tracking tool Weights and Biases (WandB)[6] [124].

[6]Weights and Biases (WandB) website: `https://wandb.ai`

Chapter 5

Results and Discussion

In this chapter the results of the model testing as outlined in the previous chapter is presented and discussed. The chapter presents the performance metrics for the final multi-label classification task for all the models used in this project, and will discuss their significance and relevance throughout the chapter.

The performance metrics for the transformer based AST and SSAST models, as well as the ResNet18 model, on the final multi-label classification task, is presented in table 5.1. Table 5.1 shows the mean and standard deviation for all metrics across training runs, as percentages to improve readability.

Metric	ResNet18	AST	SSAST
Accuracy	94.8 ± 0.5	**97.1 ± 0.4**	72.7 ± 4.8
Precision	66.9 ± 1.9	**78.9 ± 2.3**	25.2 ± 2.8
Recall	97.7 ± 0.5	**98.0 ± 0.1**	74.0 ± 7.0
F1Score	79.4 ± 1.5	**87.4 ± 1.4**	36.8 ± 1.9
meanAveragePrecision	92.8 ± 1.2	**95.7 ± 0.9**	43.2 ± 0.4
AUROC	99.2 ± 0.2	99.5 ± 0.1	82.5 ± 0.0

Table 5.1: Performance metrics of ResNet18, AST and SSAST on the GLIDER dataset

From table 5.1 it is clear that the supervised models significantly outperform the self-supervised transformer. This might indeed be expected due to the way in which the dataset splitting differs with the self-supervised model compared to the supervised models. For the self-supervised model, the amount of data available during fine-tuning is significantly lower than for the supervised models. Because we reserve 80% of the clips in the clipped dataset for self-supervised pretraining, this data is not available during fine-tuning, so as to avoid the model overfitting during fine-tuning because it would have already seen this portion of the training data. As was described in section 4.2.4, because we need both training, validation and testing subsets during fine-tuning, actually only 16% of the clips in the clipped dataset are available during training in the fine-tuning phase, as was presented in figure 4.11. This is because we reserve 20% of the clips for fine-tuning, and reserve 80% of these clips for *training* in the fine-tuning phase, resulting in 16%

of the clips being used during training in the fine-tuning phase ($80\% \cdot 20\% = 16\%$). Therefore the supervised framework has significantly fewer clips available during fine-tuning compared to both the ResNet18 and AST models, which may help explain the large difference in performance between the supervised and self-supervised models. The hypothesis that the differing dataset splits help explain the large discrepancy in performance between the supervised and self-supervised paradigms, is somewhat strengthened by the performance of the comparatively much less complex ResNet18 model relative to the SSAST.

Considering that the AST is significantly more complex than the ResNet18, in therms of the number of trainable parameters, the comparable performance for the two models is interesting. While the AST model has 87.2 million trainable parameters the ResNet18 architecture has 11.2 million trainable parameters. This makes performing inference using the ResNet18 model significantly less computationally expensive compared to the AST model. At best the almost 8-fold increase in model complexity from the ResNet18 to the AST, has yielded about 11% increase in precision, this being the metric with greatest difference between the two models. If on-board classification of marine acoustic data should be considered with glider-based Autonomous Underwater Vehicles (AUVs), these being low-power platforms intended to be deployed for up to months at a time with limited computational resources, the complexity of the classification model will be significant. Therefore it is likely more relevant for smaller models to be deployed on these platforms, as they can perform reasonably well and require significantly less computational resources for inference.

Furthermore, we can also see that precision is the least performant metric for all models. Meaning that the models struggle with false positives compared to the number of true positives at the default threshold of 0.5. However, as precision, recall, accuracy and F1 scores are calculated at a single specific threshold of 0.5, these metrics would be affected by changing the threshold value. The mean Average Precision and AUROC metrics however, are calculated based on the model predictions at different thresholds, and therefore are not affected in the same way. The mean Average Precision is calculated as the weighted mean of previsions achieved at different thresholds, with the difference in recall from the last threshold as the weight, averaged across the two classes. Meaning that the mAP combines the precision and recall values achieved at different thresholds as a single value. Similarly, the AUROC tells us the area under the curve defined by the True Positive Rate (TPR) and False Positive Rate (FPR) values achieved by the model at different thresholds [121, 125]. Meaning that neither metric depends on a single threshold value. A perfect classifier would achieve an AUROC of 1.0, whereas a random classifier would achieve an AUROC of 0.5. Therefore these metrics provide a better overall understanding of model performance. From the mAP and AUROC metrics in table 5.1, it is clear that the ResNet18 and AST models both perform quite well on the task of multi-label classification on the GLIDER dataset. From the high recall and comparatively lower precision at the default threshold, for both the ResNet18 and AST models, we can tell that both models struggle the most with the number of false positives. Meaning that the models predict clips as being anthropogenic or biophonic, when the clip does not have such a corresponding label.

This raises the issue of noisy labels in the GLIDER dataset. As only the positive sound events in the GLIDER dataset are manually labeled by Akvaplan-Niva, and because we consider any clip that does not overlap temporally with any of these labels as negative samples, there is very likely a portion of falsely negatively labeled clips. Meaning that because we have no manual labels stating that some interval of the dataset does in fact *not* contain any anthropogenic nor biophonic signals, and we consider any clips without an overlapping label as negative clips. During training, this likely leads us to present the models with clips that actually contain anthropogenic and/or biotic signals, while training the model to predict the clip as containing no such sound event. Therefore we rely on the assumption that most of the clips we consider negative instances, are indeed actually negative when applying the negative labels to the clipped dataset. Considering that the collected data span a long period of time, with consecutive recordings for hours at a time, and that the sound events we aim to detect are comparatively transient in nature, such as the call of a whale or the engine noise of a passing ship, this might indeed be the case. Meaning that because the sound events we try to detect are comparatively short, such that most clips that lack a positive manual label, might indeed not actually contain any positive sound events.

We might get a more intuitive understanding of the model performance in terms of true and false positives and negatives, by investigating the confusion matrices of the models. The target normalized confusion matrices for the AST, ResNet18 and SSAST models can be found in figures 5.1 and 5.2. The confusion matrix values for each model is first computed as the arithmetic mean of all relevant training runs performed with the model. After which the values are target normalized by dividing each the value by the sum of instances for the model in that target row. Such that the True Positive value for the ResNet18 model in figure 5.1, is calculated as the quotient of the number of True Positives, and the sum of models True Positives and False Negatives. Meaning that the True Positives is divided by the total number of positive instances. Similarly the True Negatives is computed as the quotient of True Negatives and the sum of negative target instances. This is done such that the confusion matrix values for each model is more easily comparable. Because the confusion matrix is computed using the model predictions and target values for the test dataset only, and the SSAST have significantly fewer examples available for testing during the fine-tuning stage, the scale of the confusion matrix values would be significantly different between the supervised and self-supervised models. Furthermore, the number of negative instances is far greater than the number of positive instances, such that the scale of the confusion matrix rows would differ greatly, thus worsen the readability of the figures.

From the confusion matrices, it is apparent that the performance difference between the AST and ResNet18 models is negligible. With only minor differences in the number of False Positives and True Negatives for both the biophonic and anthropogenic clips between the two models.

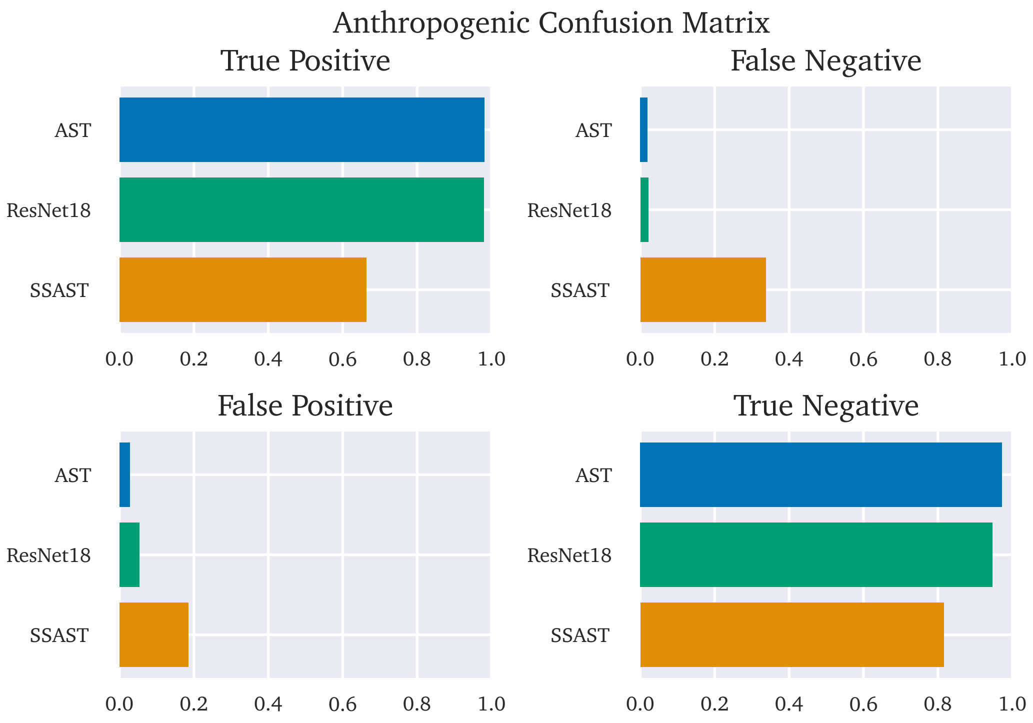

Figure 5.1: Target Normalized Confusion Matrix for anthropogenic sound events

Furthermore, the confusion matrices help make it clear that both the AST and ResNet18 models are clearly able to learn to *distinguish* clips containing anthropogenic and/or biophonic sound events from clip with no such sound events. As the percentage of false predictions (secondary diagonal) is significantly less than the percentage of true predictions (main diagonal), indicating that the models are indeed able to discern the difference between positive and negative instances. While the SSAST, although able to distinguish this to some degree, is far less capable in this regard compared to the supervised models.

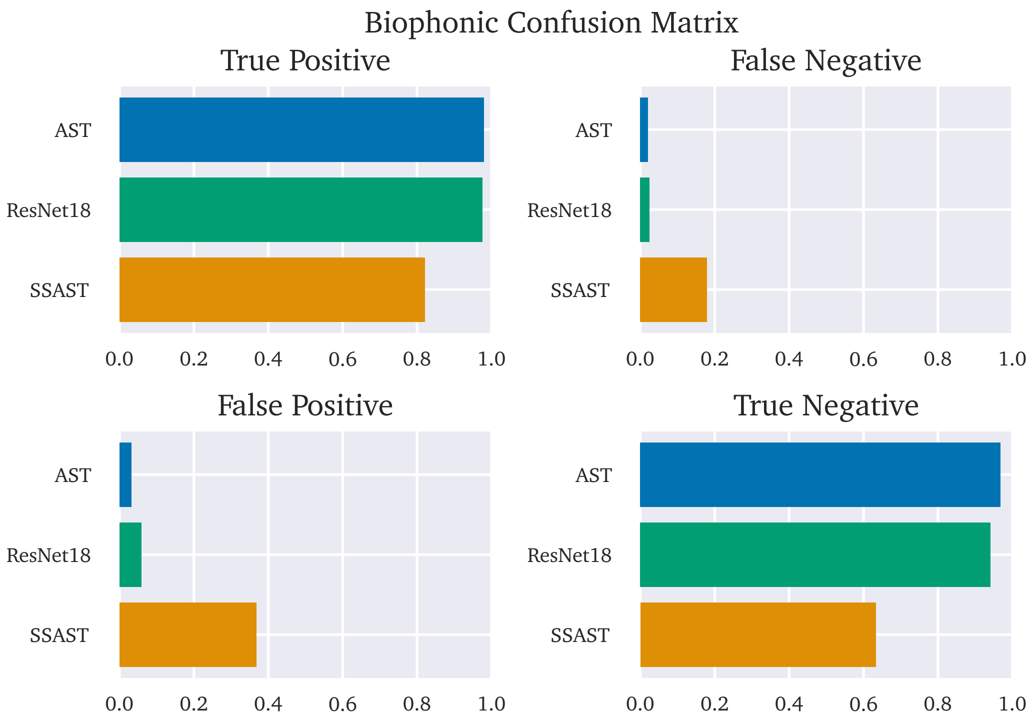

Figure 5.2: Target Normalized Confusion Matrix for biophonic sound events

Chapter 6

Conclusion

In this thesis we have explored the applicability of modern transformer neural networks in the marine audio domain, which have achieved promising results in the terrestrial audio domain. To achieve the research goals of this thesis, we implemented data pipelines for supervised and self-supervised model training paradigms, trained and evaluated the performance of one convolutional and one transformer based neural network using supervised techniques. In addition, we trained and evaluated the same transformer model using self-supervised techniques.

This project intended to answer three main research questions, regarding how deep learning based models could be applied to the GLIDER dataset, and how the performance of the supervised ResNet18 and the SSAST models compare to the supervised AST, for the task of multi-label classification of glider based marine acoustic data, to detect and classify anthropogenic and biotic sound events. We answer these questions in detail later in this chapter, and will only briefly summarize our findings here.

The details of the data pipeline described throughout this thesis, which was implemented for the training and evaluation of the models used in this work, enable us to effectively describe one of the possible methods that enable deep learning based models to be utilized with the GLIDER dataset. This pipeline involves, preparation of the raw audio and label data, including clipping, balancing of the training set, normalization and augmentation techniques, as well as a number of details that ensure the efficiency of the data pipeline during distributed training in managed high performance computing environments. While self-supervised (and unsupervised) techniques have the intuitive advantage over supervised techniques in that they do not require large amounts of manually labeled data, which can be an expensive and time-consuming effort to create, requiring expert domain knowledge to perform accurately. A trait that initially make self-supervised techniques a promising alternative in the marine audio domain, where there often is a large difference in the amount of recorded audio compared to the amount of labeled data. It is clear from the results presented in chapter 5 of this thesis, that supervised models can significantly outperform self-supervised models, provided that the amount of labeled data available for training is sufficient. Furthermore, regarding the performance difference of the two supervised models, our results show that while the ResNet18 *is* outperformed by the AST, there is only a minor difference in performance between the two models. Making the ResNet18 architecture an interesting alternative to the more complex AST for in-situ sound event detection and classification on low-power systems such as gliders. However, based on the results presented in this thesis, we can conclude that the supervised AST is arguably the more appropriate model for multi-label classification of glider-based marine acoustic data, provided sufficient computational requirements are met.

The contributions of this thesis are: 1) A quantitative comparison of the performance of the ResNet18, AST and SSAST for clip-wise multi-label classification, to detect and classify anthropogenic and biotic sound events in glider-based marine acoustic data which was collected using a (comparatively) high sampling rate. 2) A thorough understanding of *one possible* data processing pipeline that enables us to utilize the GLIDER dataset with deep learning based classification models. 3) A publicly available code repository[1], containing an implementation of the above mentioned data processing pipeline, and the trained parameters for the models evaluated in this thesis.

6.1 Research Questions

In this section the research questions of this thesis are answered.

RQ1 *How can deep learning models be applied for multi-label classification of anthropogenic and biophonic sound events with the GLIDER dataset?*

The answer to this question is primarily found through the implementation of a suitable data pipeline for training, validation and testing of the models used in this project, specifically regarding some of the theoretical background relating to the audio domain and machine learning, as well as some of the implementation details of the pipeline as described in chapter 4 of this thesis. In the following paragraph we will reiterate on these and the reasoning behind them, in answering this question.

Acoustics in the marine audio domain pose a number of challenges to traditional audio classification models, notably the attributes of water as an acoustic environment that is highly affected by parameters such as temperature, pressure (depth) and salinity. An effect that is exacerbated for marine acoustic data collected using mobile recording platforms that traverse the water column, such as gliders. As well as the target sound sources in marine environment often utilizing frequencies far higher than those typically found in terrestrial environments, which make it necessary for marine acoustic data to be collected at comparatively far higher sampling rates. This in turn potentially causes limitations or challenges

[1]Github repository: `https://github.com/MartinMoan/TDT4900-Noise-in-the-Sea-Source`

in using the same parameters in computing the time-frequency representations used for terrestrial audio data, as the computational complexity of typical deep learning architectures increase with the increase of input dimensions. However, based on the results presented in chapter 5, all of the models evaluated in this work seem to perform reasonably well for glider based marine acoustic data, and does not seem to have much issue in detecting anthropogenic and biophonic signals in the acoustic data. Similarly, as the computational requirements increase with increasing input spectrogram dimensions with these models, the duration of the input to these models need to be kept reasonable. One solution to this problem is to clip the underlying audio data, to ensure that the time-dimension of the spectrograms be kept manageable. This however, introduces a number of requirements to the implementation of the data pipeline used (especially) when training. Because the labels of the GLIDER dataset are not balanced, and we need to ensure that the part of the data used during training, is balanced. We need to find the overlapping labels for every clip within the dataset, such that we may ensure that the number of clips per label is balanced during training. However, as the labels of the GLIDER dataset describe a class label, start- and ending timestamps, we also need to know the starting and ending timestamps of every clip. Such that we may find the temporally overlapping labels to every clip. But because the filename information of the recordings in the GLIDER dataset only provide information as to its starting timestamp, and because some of the recordings are not of the expected 10 minute duration, we need it iterate over all the recordings of the dataset to compute their ending timestamps based on their duration. Keeping in mind that this involves loading every recording into memory and checking the number of samples within it, and not relying on the number of samples stated in the binary header information of the recordings, as this header information is in some cases not correct. After we have extracted the starting and ending timestamps for every recording, we can effectively apply clipping to the recordings, by computing the offset at which each clip should start relative to its recording. This ensures that the time-dimension of the spectrograms can be kept reasonably low for computational efficiency during inference and training, while also enabling us to compute the number of clips that are available during training, and their clip-wise starting and ending timestamps. The starting and ending timestamp of the clips can then be used to find all the labels which overlap with any given clip, such that we are able to balance the clips according to their labels for the training set. Not only does this enable the training set to be balanced according to clip labels, but also ensures that we can utilize effective multi-process dataloading with PyTorch by implementing the dataset as a Map-style dataset rather than an Iterable-style dataset without replicating the dataset instance across processes/GPUs.

These are all details that help describe the complexity in implementing an efficient data pipeline for audio datasets in general, and for the GLIDER dataset specifically. Ensuring the efficiency of the data pipeline is necessary for parallel computation with deep neural networks, so as not to introduce any unnecessary bottlenecks during training, and thereby keeping training time reasonable. As we have seen from chapter 4, the way in which the audio data of the GLIDER dataset is created, and metadata formatted and stored, introduces a number of challenges and requirements in how we can effectively apply the dataset with modern deep learning techniques.

RQ2 *How does self-supervised pretraining of the AST (SSAST) affect the performance of multi-label classification of glider based marine acoustic data?*

As we have seen from the results presented in chapter 5, the self-supervised pretraining of the SSAST with the GLIDER dataset, reduces the performance of the model after fine-tuning for the final prediction task compared to the supervised framework. The supervised AST achieves an F1-score of approximately 87.4 mAp of approximately 95.7 and an AUROC of approximately 99.5, which significantly outperforms the SSAST for all metrics, only achieving F1-score, mAp and AUROC of approximately 36.8, 43.2 and 82.5 respectively. One possible explanation to this discrepancy might be the differing dataset splits applied in this work for the supervised and self-supervised frameworks with the GLIDER dataset. However, as the self-supervised framework achieves a decent AUROC of 82.5, it can still be suitable for datasets with less labeled data. This also raises the question of how to interpret the labels of clips which do not overlap with any labels with the GLIDER dataset, were the lack of label does not unequivocally correspond to no sound event happening within the clip. By considering these clips as negative instances, as we do in this project, we thereby increase the amount of noise within the labels of the GLIDER dataset. To determine whether the discrepancy in performance between the supervised and self-supervised frameworks is a result of such label noise, the differing dataset splits, or something else entirely, further research is necessary.

RQ3 *How does the performance of the CNN-based ResNet18 model compare to the Audio Spectrogram Transformer (AST) and Self-Supervised Audio Spectrogram Transformer (SSAST) for the task of multi-label classification of glider-based marine acoustic data?*

For the multi-label classification task performed in this work using the GLIDER dataset, the ResNet18 achieves an F1-score, mAp and AUROC of 79.4, 92.8 and 99.2 respectively. While being outperformed by the AST for every considered metric in this work, the ResNet18 still provides decent results for the classification task, while being considerably smaller in terms of trainable parameters compared to the AST. This arguably makes the ResNet18 architecture a more interesting alternative to more complex models for online audio classification on low-powered deployed systems in the future. It is clear however that the ResNet18 has poor precision of approximately 66.9 at the default classification threshold, which is likely reducing the performance of the model in regard to the other metrics. However, for sound event detection and classification performed after recovery of glider-based recording platforms and their acoustic data, the supervised AST is arguably the most appropriate, as this is the best performing model overall among the three evaluated in this thesis.

6.2 Limitations of this work

Some limitations in this work are the lack of hyperparameter optimization and cross validation. The hyperparameters used in this work, such as number of mel filters, STFT window size and hop length, as well as optimizer hyperparameters was all chosen either based on values provided in the relevant model paper for the AST, SSAST and ResNet18, or by informal experimentation. But these should rather be based on hyperparameter optimization performed on the validation set. Furthermore, cross validation should arguably also be utilized, as the model performance is subject to the data presented to the model during training and testing. The clips that are reserved for each subset is chosen semi-randomly, by random selection with seed hardcoded to ensure repeatability should it be necessary. This ensures that the dataset objects that are instantiated among the GPU processes on the computing cluster reserves the same clips for training, validation and testing. If this is not ensured, one GPU process might put a given clip in the training set, while another GPU process put the same clip into the test set. Because the model gradients are synced among the GPUs after the backward pass, this would provide incorrect testing results, because the model would be tested on a clip it would have seen during training. Utilizing cross validation would allow us to gain a more thorough understanding of the model performance, but has not been performed in this work due to time constraints and complexity in implementing this correctly using distributed training.

6.3 Future Work

In this section some suggestions for future improvements to the current GLIDER dataset classifiers and to future work in the marine glider based audio domain in general are presented.

FW1 *Spectrogram frame-based embedding*

The original AST (and SSAST) papers [14, 15] both utilize the same patching operation as originally proposed by Dosovitskiy *et al.* in [36]. By breaking the input image, or spectrogram, into patches and linearly projecting the flattened patches, to convert the input image/spectrogram into a sequence of embedding vectors that can be utilized by the transformer architecture. For image classification tasks this does seem reasonable, as there is no inherent ordering to the patches of an image. However, this is not the case for time-frequency representations of audio, in which a definite temporal order exists within the spectrograms, which typically are read from the start to the end of the recording left-to-right. Therefore we suggest that future work using the transformer architecture for the audio domain evaluate the effectiveness of alternative embedding techniques, that ensure that the input embedding vectors represent some meaningful information of its origin spectrogram along the time dimension. One example might be to compute the embedding vectors from a single, or some small number, of time-frames of the original spectrogram. Thus forgoing the 16x16 image patching technique utilized in [36] and adopted by the AST and SSAST [14, 15]. Our hypothesis is that because such embeddings would include information about the entirety of the frequency spectrum at any point in time (or short duration), the transformer might be more easily able to learn the temporal structure of sound events. With the currently applied 16x16 patching technique, the model must not only learn the temporal structure of the sound events but must also learn to "stitch together" the order of the patch embeddings. Applying this type of frame-based embedding might indeed be considered introducing an inductive bias to the model, similar to the translational equivariance of convolutional models, but we consider evaluating such an embedding technique to be an interesting contribution to machine learning in the audio domain in general as it poses an arguably more intuitive approach in transforming time-frequency representations to embedding sequences for transformers. Using such a frame-based embedding technique would still require some positional encoding to be applied, as the self-attention mechanism does not include such information by default.

FW2 *Cepstral features together with spectrogram input*

Recently, multi-modal neural networks has seen increased use [126–128]. With models that can process not only audio data but, for example, video data simultaneously. In this regard we suggest future work investigate how we can utilize similar techniques to enable deep learning based audio classification models to utilize both time-frequency representations as well as other audio features, such as cepstral features, as joint input. We suggest cepstral features here with the hypothesis that deep learning based models might be better able to learn periodic structures in the data by utilizing the signal cepstrum directly, rather than learning these periodic structures from the temporal structure of the spectrogram itself [129].

FW3 *Lightweight classifiers*

We suggest that future work in the audio domain, particularly in the marine audio domain using autonomous low-power recording platforms such as gliders, investigate the applicability of smaller neural network classifiers for low-power deployed systems. These platforms, while being able to collect large volumes of acoustic data, spanning large geographic areas and temporal ranges, do not currently enable real-time reporting of acoustic events, but rather require that the audio data be processed after collection of the vehicle. Smaller neural networks, such as the ResNet18 [72] or MobileNets [130], due to their relatively small size, might be more applicable for on-board sound event detection and classification on deployed AUVs where there are limitations to the acceptable computational load and energy consumption of the classifiers. We believe that the performance of the comparatively smaller ResNet18 model in relation to the AST presented in this thesis, provides some evidence to the hypothesis that these models might open the door to on-board sound event detection and classification on deployed low-power systems. This could make low-power AUVs far more applicable in long term passive acoustic monitoring, enabling these platforms to report sound events while deployed, by transmitting only the acoustic data containing relevant sound events.

FW4 *Multi-sensor data for audio classification*

The GLIDER dataset contains non-acoustic data collected using various sensors during deployment, such as depth, salinity, temperature and geographical coordinates of the recording platform. In this project this data has not been used, as we have primarily been interested in using deep learning based models on the acoustic data directly. However, this data might be useful for the classification task. We know that the properties of acoustics in marine environments change based on temperature, salinity and pressure, as well as by objects in the environment such as the seabed itself and geological structures. We therefore suggest that future work investigate how this data can be used, in conjunction with time-frequency based audio features, to perform multi-label classification of the acoustic data of the GLIDER dataset. We suggest this with the hypothesis that the model might be able to learn the dependency between the recording environment and the sound events within the time-frequency audio representations, using the temperature, salinity and depth of the surrounding water, to increase the model performance. This suggestion can also tie in with the suggestion from FW2 regarding multi-modal models. Using this data, we could estimate the speed of sound in the immediately surrounding environment [131, 132] and similarly provide this alongside the time-frequency representation. Another possibility is to use the coordinates of the glider, and other sensor data, together with openly accessible data that describes environmental properties such as ocean depth, bottom sediments and geological structures of the environment, in the audio classification task. We suggest this with the hypothesis that such multi-modal input could help the model learn the relationship between the surrounding ocean environment and the recorded audio, such that the model might be able to account for effects such as multipath sound propagation and warping effects that can occur in marine acoustics due to the recording environment, when performing the sound event detection and classification task.

FW5 *Transformers for other marine audio datasets*

Finally, we propose that future work investigate the performance of transformer based models for other marine audio datasets. We suggest evaluating such models on audio datasets collected using differing recording platforms than the

GLIDER dataset, for example using ship-towed hydrophone arrays or stationary recording devices. We suggest this as the results of this thesis clearly show that the convolutional models, which have been pervasive in the marine audio domain [12], are not indispensable for classification of glider-based marine acoustic data, and can be outperformed by transformer based models. For this reason, we consider investigating whether transformer based models also outperform convolutional models for other marine audio datasets an interesting prospect.

Bibliography

[1] M. Moan, 'Noise in the Sea - Deep Learning for Sound Event Detection and Classification of Marine Acoustic Data,' Specialization Project Final Report, Dec. 2021.

[2] Merriam-Webster.com Dictionary, *Anthropogenic*, Accessed 30 Jun. 2021. [Online]. Available: `https://www.merriam-webster.com/dictionary/anthropogenic`.

[3] Wikipedia contributors, *Bioacoustics — Wikipedia, the free encyclopedia*, `https://en.wikipedia.org/w/index.php?title=Bioacoustics&oldid=1093854950`, [Online; accessed 30-June-2022], 2022.

[4] Wiktionary, *Biophony — wiktionary, the free dictionary*, [Online; accessed 30-June-2022], 2021. [Online]. Available: `%5Curl%7Bhttps://en.wiktionary.org/w/index.php?title=biophony&oldid=62198534%7D`.

[5] Wikipedia contributors, *Animal echolocation — Wikipedia, the free encyclopedia*, [Online; accessed 17-June-2022], 2022. [Online]. Available: `https://en.wikipedia.org/w/index.php?title=Animal_echolocation&oldid=1091108629`.

[6] Wikipedia contributors, *Mel scale — Wikipedia, the free encyclopedia*, Accessed 16 Jun. 2021., 2021. [Online]. Available: `https://en.wikipedia.org/w/index.php?title=Mel_scale&oldid=1061063889`.

[7] Wikipedia contributors, *Ocean networks canada — Wikipedia, the free encyclopedia*, [Online; accessed 19-June-2022], 2021. [Online]. Available: `https://en.wikipedia.org/w/index.php?title=Ocean_Networks_Canada&oldid=1043810824`.

[8] The Editors of Encyclopaedia Britannica, *Sofar channel*, `https://www.britannica.com/art/octave-music`, [Online; accessed 24 June 2022], Apr. 2017.

[9] Wikipedia contributors, *Pitch (music) — Wikipedia, the free encyclopedia*, `https://en.wikipedia.org/w/index.php?title=Pitch_(music)&oldid=1076385976`, [Online; accessed 30-June-2022], 2022.

[10] A. B. Yoo, M. A. Jette and M. Grondona, 'Slurm: Simple linux utility for resource management,' in *Job Scheduling Strategies for Parallel Processing*, D. Feitelson, L. Rudolph and U. Schwiegelshohn, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 44–60, ISBN: 978-3-540-39727-4.

[11] Merriam-Webster.com Dictionary, *Soniferous*. Accessed 16 Jun. 2021. [Online]. Available: `https://www.merriam-webster.com/dictionary/soniferous`.

[12] F. Frazao, B. Padovese and O. S. Kirsebom, *Workshop report: Detection and classification in marine bioacoustics with deep learning*, 2020. arXiv: `2002.08249 [eess.AS]`.

[13] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser and I. Polosukhin, *Attention is all you need*, 2017. DOI: `10.48550/ARXIV.1706.03762`. [Online]. Available: `https://arxiv.org/abs/1706.03762`.

[14] Y. Gong, Y.-A. Chung and J. Glass, 'AST: Audio Spectrogram Transformer,' in *Proc. Interspeech 2021*, 2021, pp. 571–575. DOI: `10.21437/Interspeech.2021-698`.

[15] Y. Gong, C.-I. J. Lai, Y.-A. Chung and J. Glass, 'SSAST: Self-Supervised Audio Spectrogram Transformer,' *arXiv preprint arXiv:2110.09784*, 2021.

[16] C. Erbe, C. Reichmuth, K. Cunningham, K. Lucke and R. Dooling, 'Communication masking in marine mammals: A review and research strategy,' *Marine Pollution Bulletin*, vol. 103, no. 1, pp. 15–38, 2016, ISSN: 0025-326X. DOI: `https://doi.org/10.1016/j.marpolbul.2015.12.007`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S0025326X15302125`.

[17] C. Erbe, S. A. Marley, R. P. Schoeman, J. N. Smith, L. E. Trigg and C. B. Embling, 'The effects of ship noise on marine mammals—a review,' *Frontiers in Marine Science*, vol. 6, 2019, ISSN: 2296-7745. DOI: `10.3389/fmars.2019.00606`. [Online]. Available: `https://www.frontiersin.org/article/10.3389/fmars.2019.00606`.

[18] C. Peng, X. Zhao and G. Liu, 'Noise in the sea and its impacts on marine organisms,' *International Journal of Environmental Research and Public Health*, vol. 12, pp. 12 304–12 323, Sep. 2015. DOI: `10.3390/ijerph121012304`.

[19] M. K. Pine, K. Nikolich, B. Martin, C. Morris and F. Juanes, 'Assessing auditory masking for management of underwater anthropogenic noise,' *The Journal of the Acoustical Society of America*, vol. 147, no. 5, pp. 3408–3417, 2020. DOI: `10.1121/10.0001218`. eprint: `https://doi.org/10.1121/10.0001218`. [Online]. Available: `https://doi.org/10.1121/10.0001218`.

[20] M. André, M. van der Schaar, S. Zaugg, L. Houégnigan, A. Sánchez and J. Castell, 'Listening to the deep: Live monitoring of ocean noise and cetacean acoustic signals,' *Marine Pollution Bulletin*, vol. 63, no. 1, pp. 18–26, 2011, Cetaceans and military sonar: A need for better management, ISSN: 0025-326X. DOI: `https://doi.org/10.1016/j.marpolbul.2011.04.038`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S0025326X11002414`.

[21] S. Zaugg, M. van der Schaar, L. Houégnigan, C. Gervaise and M. André, 'Real-time acoustic classification of sperm whale clicks and shipping impulses from deep-sea observatories,' *Applied Acoustics*, vol. 71, no. 11, pp. 1011–1019, 2010, Proceedings of the 4th International Workshop on Detection, Classification and Localization of Marine Mammals Using Passive Acoustics and 1st International Workshop on Density Estimation of Marine Mammals Using Passive Acoustics, ISSN: 0003-682X. DOI: `https://doi.org/10.1016/j.apacoust.2010.05.005`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S0003682X1000112X`.

[22] M. Bittle and A. Duncan, 'A review of current marine mammal detection and classification algorithms for use in automated passive acoustic monitoring,' in *Proceedings of Acoustics*, Citeseer, vol. 2013, 2013.

[23] M. F. Baumgartner, K. M. Stafford, P. Winsor, H. Statscewich and D. M. Fratantoni, 'Glider-based passive acoustic monitoring in the arctic,' *Marine Technology Society Journal*, vol. 48, no. 5, 2014.

[24] R. Davis, M. Baumgartner, A. Comeau, D. Cunningham, K. Davies, A. Furlong, H. Johnson, S. L'Orsa, T. Ross, C. Taggart and F. Whoriskey, 'Tracking whales on the scotian shelf using passive acoustic monitoring on ocean gliders,' in *OCEANS 2016 MTS/IEEE Monterey*, 2016, pp. 1–4. DOI: `10.1109/OCEANS.2016.7761461`.

[25] M. F. Baumgartner, D. M. Fratantoni, T. P. Hurst, M. W. Brown, T. V. N. Cole, S. M. Van Parijs and M. Johnson, 'Real-time reporting of baleen whale passive acoustic detections from ocean gliders,' *The Journal of the Acoustical Society of America*, vol. 134, no. 3, pp. 1814–1823, 2013. DOI: `10.1121/1.4816406`. eprint: `https://doi.org/10.1121/1.4816406`. [Online]. Available: `https://doi.org/10.1121/1.4816406`.

[26] L. Camus, H. Andrade, A. S. Aniceto, M. Aune, K. Bandara, S. L. Basedow, K. H. Christensen, J. Cook, M. Daase, K. Dunlop, S. Falk-Petersen, P. Fietzek, G. Fonnes, P. Ghaffari, G. Gramvik, I. Graves, D. Hayes, T. Langeland, H. Lura, T. Kristiansen, O. A. Nøst, D. Peddie, J. Pederick, G. Pedersen, A. K. Sperrevik, K. Sørensen, L. Tassara, S. Tjøstheim, V. Tverberg and S. Dahle, 'Autonomous surface and underwater vehicles as effective ecosystem monitoring and research platforms in the arctic—the glider project,' *Sensors*, vol. 21, no. 20, 2021, ISSN: 1424-8220. DOI: `10.3390/s21206752`. [Online]. Available: `https://www.mdpi.com/1424-8220/21/20/6752`.

[27] S. Kahl, T. Wilhelm-Stein, H. Klinck, D. Kowerko and M. Eibl, *Recognizing birds from sound - the 2018 birdclef baseline system*, 2018. DOI: 10.48550/ARXIV.1804.07177. [Online]. Available: https://arxiv.org/abs/1804.07177.

[28] N. Turpault, R. Serizel, A. Parag Shah and J. Salamon, 'Sound event detection in domestic environments with weakly labeled data and soundscape synthesis,' working paper or preprint, Jun. 2019. [Online]. Available: https://hal.inria.fr/hal-02160855.

[29] M. F. Baumgartner and S. E. Mussoline, 'A generalized baleen whale call detection and classification system,' *The Journal of the Acoustical Society of America*, vol. 129, no. 5, pp. 2889–2902, 2011. DOI: 10.1121/1.3562166. eprint: https://doi.org/10.1121/1.3562166. [Online]. Available: https://doi.org/10.1121/1.3562166.

[30] B. Mishachandar and S. Vairamuthu, 'Diverse ocean noise classification using deep learning,' *Applied Acoustics*, vol. 181, p. 108 141, 2021, ISSN: 0003-682X. DOI: https://doi.org/10.1016/j.apacoust.2021.108141. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0003682X21002358.

[31] S. Sheng, H. Yang, L. Junhao, G. Xu and M. Sheng, 'Auditory inspired convolutional neural networks for ship type classification with raw hydrophone data,' *Entropy*, vol. 20, p. 990, Dec. 2018. DOI: 10.3390/e20120990.

[32] H. A. Garcia, T. Couture, A. Galor, J. M. Topple, W. Huang, D. Tiwari and P. Ratilal, 'Comparing performances of five distinct automatic classifiers for fin whale vocalizations in beamformed spectrograms of coherent hydrophone array,' *Remote Sensing*, vol. 12, no. 2, 2020, ISSN: 2072-4292. DOI: 10.3390/rs12020326. [Online]. Available: https://www.mdpi.com/2072-4292/12/2/326.

[33] C. McQuay, F. Sattar and P. F. Driessen, 'Deep learning for hydrophone big data,' in *2017 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM)*, 2017, pp. 1–6. DOI: 10.1109/PACRIM.2017.8121894.

[34] Y. Shiu, K. J. Palmer, M. A. Roch, E. Fleishman, X. Liu, E.-M. Nosal, T. Helble, D. Cholewiak, D. Gillespie and H. Klinck, 'Deep neural networks for automated detection of marine mammal species,' *Scientific Reports*, vol. 10, no. 1, p. 607, Jan. 2020. DOI: 10.1038/s41598-020-57549-y. [Online]. Available: https://doi.org/10.1038/s41598-020-57549-y.

[35] E. L. Ferguson, 'Multitask convolutional neural network for acoustic localization of a transiting broadband source using a hydrophone array,' *The Journal of the Acoustical Society of America*, vol. 150, no. 1, pp. 248–256, 2021. DOI: 10.1121/10.0005516. eprint: https://doi.org/10.1121/10.0005516. [Online]. Available: https://doi.org/10.1121/10.0005516.

[36] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit and N. Houlsby, *An image is worth 16x16 words: Transformers for image recognition at scale*, 2020. DOI: 10.48550/ARXIV.2010.11929. [Online]. Available: https://arxiv.org/abs/2010.11929.

[37] NOAA Pacific Islands Fisheries Science Center, *Hawaiian Islands Cetacean and Ecosystem Assessment Survey (HICEAS) towed array data. Edited and annotated for the 9th International Workshop on Detection, Classification, Localization, and Density Estimation of Marine Mammals Using Passive Acoustics (DCLDE 2022)*, [Accessed 19. Jun. 2022.], 2022. [Online]. Available: https://doi.org/10.25921/e12p-gj65.

[38] E. Cakir, T. Heittola, H. Huttunen and T. Virtanen, 'Polyphonic sound event detection using multi label deep neural networks,' in *2015 International Joint Conference on Neural Networks (IJCNN)*, 2015, pp. 1–7. DOI: 10.1109/IJCNN.2015.7280624.

[39] E. Cakir, G. Parascandolo, T. Heittola, H. Huttunen and T. Virtanen, 'Convolutional recurrent neural networks for polyphonic sound event detection,' *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 6, pp. 1291–1303, Jun. 2017, ISSN: 2329-9304. DOI: 10.1109/taslp.2017.2690575. [Online]. Available: http://dx.doi.org/10.1109/TASLP.2017.2690575.

[40] Wikipedia contributors, *Sound — Wikipedia, the free encyclopedia*, https://en.wikipedia.org/w/index.php?title=Sound&oldid=1092775001, [Online; accessed 23-June-2022], 2022.

[41] R. E. Berg, *Sound*, Accessed 16 Jun. 2021., 2020. [Online]. Available: https://www.britannica.com/science/sound-physics.

[42] Wikipedia contributors, *Hydrophone — Wikipedia, the free encyclopedia*, https://en.wikipedia.org/w/index.php?title=Hydrophone&oldid=1089348984, [Online; accessed 23-June-2022], 2022.

[43] Wikipedia contributors, *Nyquist–shannon sampling theorem — Wikipedia, the free encyclopedia*, https://en.wikipedia.org/w/index.php?title=Nyquist%E2%80%93Shannon_sampling_theorem&oldid=1091151231, [Online; accessed 21-June-2022], 2022.

[44] Wikipedia contributors, *44,100 hz — Wikipedia, the free encyclopedia*, https://en.wikipedia.org/w/index.php?title=44,100_Hz&oldid=1092674793, [Online; accessed 23-June-2022], 2022.

[45] D. Purves, G. J. Augustine, D. Fitzpatrick, W. C. Hall, A.-S. LaMantia, R. D. Mooney, M. L. Platt and L. E. White, Eds., *Neuroscience*, en, 6th ed. Sinauer Associates, Jan. 2018, ISBN: 9781605353807.

[46] Wikipedia contributors, *Short-time fourier transform — Wikipedia, the free encyclopedia*, https://en.wikipedia.org/w/index.php?title=Short-time_Fourier_transform&oldid=1076715527, [Online; accessed 21-June-2022], 2022.

[47] Wikipedia contributors, *Fourier transform — Wikipedia, the free encyclopedia*, https://en.wikipedia.org/w/index.php?title=Fourier_transform&oldid=1094517380, [Online; accessed 23-June-2022], 2022.

[48] T. Giannakopoulos and A. Pikrakis, 'Chapter 3 - signal transforms and filtering essentials,' in *Introduction to Audio Analysis*, T. Giannakopoulos and A. Pikrakis, Eds., Oxford: Academic Press, 2014, pp. 33–57, ISBN: 978-0-08-099388-1. DOI: https://doi.org/10.1016/B978-0-08-099388-1.00003-0. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780080993881000030.

[49] Wikipedia contributors, *Complex number — Wikipedia, the free encyclopedia*, https://en.wikipedia.org/w/index.php?title=Complex_number&oldid=1096420352, [Online; accessed 4-July-2022], 2022.

[50] Wikipedia contributors, *Spectral leakage — Wikipedia, the free encyclopedia*, https://en.wikipedia.org/w/index.php?title=Spectral_leakage&oldid=1070219872, [Online; accessed 23-June-2022], 2022.

[51] Wikipedia contributors, *Window function — Wikipedia, the free encyclopedia*, https://en.wikipedia.org/w/index.php?title=Window_function&oldid=1091162436, [Online; accessed 30-June-2022], 2022.

[52] B. McFee, A. Metsai, M. McVicar, S. Balke, C. Thomé, C. Raffel, F. Zalkow, A. Malek, Dana, K. Lee, O. Nieto, D. Ellis, J. Mason, E. Battenberg, S. Seyfarth, R. Yamamoto, viktorandreevichmorozov, K. Choi, J. Moore, R. Bittner, S. Hidaka, Z. Wei, nullmightybofo, D. Hereñú, F.-R. Stöter, P. Friesch, A. Weiss, M. Vollrath, T. Kim and Thassilo, *Librosa/librosa: 0.8.1rc2*, version 0.8.1rc2, May 2021. DOI: 10.5281/zenodo.4792298. [Online]. Available: https://doi.org/10.5281/zenodo.4792298.

[53] G. Van Rossum and F. L. Drake Jr, *Python reference manual*. Centrum voor Wiskunde en Informatica Amsterdam, 1995.

[54] P. Upadhyaya, O. Farooq and M. R. Abidi, 'Mel scaled m-band wavelet filter bank for speech recognition,' *International Journal of Speech Technology*, vol. 21, no. 4, pp. 797–807, 2018. DOI: 10.1007/s10772-018-9545-2. [Online]. Available: https://doi.org/10.1007/s10772-018-9545-2.

[55] A. Mesaros, T. Heittola, T. Virtanen and M. D. Plumbley, 'Sound event detection: A tutorial,' *IEEE Signal Processing Magazine*, vol. 38, no. 5, pp. 67–83, Sep. 2021, ISSN: 1558-0792. DOI: 10.1109/msp.2021.3090678. [Online]. Available: http://dx.doi.org/10.1109/MSP.2021.3090678.

[56] Engineering ToolBox, *Sound intensity, power and pressure levels*, `https://www.engineeringtoolbox.com/sound-power-intensity-pressure-d_57.html`, [Accessed 24. Jun. 2022], 2003.

[57] National Physical Laboratory, *Sound pressure*, Accessed 16 Jun. 2021., 2018. [Online]. Available: `http://resource.npl.co.uk/acoustics/techguides/concepts/spl.html`.

[58] Discovery of Sound in the Sea (DOSITS), *Introduction to decibels*, `https://dosits.org/science/advanced-topics/introduction-to-decibels/`, [Online; accessed 16 Jun. 2021].

[59] Discovery of Sound in the Sea (DOSITS), *Sound speed minimum*, `https://dosits.org/science/movement/sofar-channel/sound-speed-minimum/`, [Online; accessed 21. June 2022].

[60] Wikipedia contributors, *Sofar channel — Wikipedia, the free encyclopedia*, `https://en.wikipedia.org/w/index.php?title=SOFAR_channel&oldid=1091065307`, [Online; accessed 23-June-2022], 2022.

[61] Discovery of Sound in the Sea (DOSITS), *Sound travel in the sofar channel*, `https://dosits.org/science/advanced-topics/introduction-to-decibels/`, [Online; accessed 23 Jun. 2021].

[62] The Editors of Encyclopaedia Britannica, *Sofar channel*, `https://www.britannica.com/science/SOFAR-channel`, [Online; accessed 23 June 2022], Sep. 2012.

[63] T. Xu and L. Xu, 'Chapter 3 - digital underwater acoustic communication signal processing,' in *Digital Underwater Acoustic Communications*, T. Xu and L. Xu, Eds., Academic Press, 2017, pp. 145–199, ISBN: 978-0-12-803009-7. DOI: `https://doi.org/10.1016/B978-0-12-803009-7.00003-9`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/B9780128030097000039`.

[64] E. L. Ferguson, S. B. Williams and C. T. Jin, *Sound source localization in a multipath environment using convolutional neural networks*, 2017. DOI: `10.48550/ARXIV.1710.10948`. [Online]. Available: `https://arxiv.org/abs/1710.10948`.

[65] B. A. Tan, M. Motani, M. Chitre and S. Quek, 'Multichannel communication based on adaptive equalization in very shallow water acoustic channels,' Nov. 2006.

[66] A. Zhang, Z. C. Lipton, M. Li and A. J. Smola, 'Dive into deep learning,' *arXiv preprint arXiv:2106.11342*, 2021.

[67] Z. Lin, M. Feng, C. N. d. Santos, M. Yu, B. Xiang, B. Zhou and Y. Bengio, *A structured self-attentive sentence embedding*, 2017. DOI: `10.48550/ARXIV.1703.03130`. [Online]. Available: `https://arxiv.org/abs/1703.03130`.

[68] Wikipedia contributors, *Softmax function — Wikipedia, the free encyclopedia*, [Online; accessed 6-June-2022], 2022. [Online]. Available: `https://en.wikipedia.org/w/index.php?title=Softmax_function&oldid=1091841696`.

[69] J. Devlin, M.-W. Chang, K. Lee and K. Toutanova, *Bert: Pre-training of deep bidirectional transformers for language understanding*, 2018. DOI: `10.48550/ARXIV.1810.04805`. [Online]. Available: `https://arxiv.org/abs/1810.04805`.

[70] A. v. d. Oord, Y. Li and O. Vinyals, *Representation learning with contrastive predictive coding*, 2018. DOI: `10.48550/ARXIV.1807.03748`. [Online]. Available: `https://arxiv.org/abs/1807.03748`.

[71] C. Bergler, H. Schröter, R. X. Cheng, V. Barth, M. Weber, E. Nöth, H. Hofer and A. Maier, 'Orca-spot: An automatic killer whale sound detection toolkit using deep learning,' *Scientific Reports*, vol. 9, no. 1, p. 10 997, Jul. 2019. DOI: `10.1038/s41598-019-47335-w`. [Online]. Available: `https://doi.org/10.1038/s41598-019-47335-w`.

[72] K. He, X. Zhang, S. Ren and J. Sun, 'Deep residual learning for image recognition,' in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778. DOI: `10.1109/CVPR.2016.90`.

[73] S. Ness, 'The orchive: A system for semi-automatic annotation and analysis of a large collection of bioacoustic recordings,' University of Victoria, 3800 Finnerty Road, Victoria, British Columbia, Canada, V8P 5C2, Dec. 2013.

[74] OrcaLab, *OrcaLab - a whale research station on Hanson Island*, Apr. 2019. [Online]. Available: `http://orcalab.org`.

[75] D. Gillespie, *DCLDE 2013 Workshop dataset*, 2019. [Online]. Available: `https://doi.org/10.17630/62c3eebc-5574-4ec0-bfef-367ad839fe1a`.

[76] Y. Lecun and Y. Bengio, 'Convolutional networks for images, speech, and time-series,' English (US), in *The handbook of brain theory and neural networks*, M. Arbib, Ed. MIT Press, 1995.

[77] K. Simonyan and A. Zisserman, *Very deep convolutional networks for large-scale image recognition*, 2014. DOI: `10.48550/ARXIV.1409.1556`. [Online]. Available: `https://arxiv.org/abs/1409.1556`.

[78] Y. Xu, Q. Kong, Q. Huang, W. Wang and M. D. Plumbley, *Convolutional gated recurrent neural network incorporating spatial features for audio tagging*, 2017. DOI: `10.48550/ARXIV.1702.07787`. [Online]. Available: `https://arxiv.org/abs/1702.07787`.

[79] M. Pourhomayoun, P. Dugan, M. Popescu, D. Risch, H. Lewis and C. Clark, 'Classification for Big Dataset of Bioacoustic Signals Based on Human Scoring System and Artificial Neural Network,' *arXiv e-prints*, arXiv:1305.3633, arXiv:1305.3633, May 2013. arXiv: `1305.3633 [cs.CV]`.

[80] M. Poupard, P. Best, J. Schlüter, J.-M. Prévot, H. Symonds, P. Spong and H. Glotin, 'Deep Learning for Ethoacoustics of Orcas on three years pentaphonic continuous recording at Orcalab revealing tide, moon and diel effects,' in *OCEANS*, Marseille, France, Jun. 2019. [Online]. Available: https://hal.archives-ouvertes.fr/hal-02445426.

[81] M. Poupard, J. Schlüter, M. Ferrari, P. Astruch, T. Schohn, E. Rouanet, A. Goujard, A. Lyonnet, P. Giraudet, V. Barchasz, V. Gies, P. Best, J.-M. Dominici, T. Lengagne, T. Soriano and H. Glotin, 'Passive acoustics to monitor flagship species near boat traffic in the Unesco World Heritage natural Reserve of Scandola,' in *Planning, Nature and Ecosystem Services*, C. G. bibinitperiod C. Zoppi, Ed., FedOAPress, 2019, pp. 260–270. [Online]. Available: https://hal.archives-ouvertes.fr/hal-02398140.

[82] T. Grill and J. Schlüter, 'Two convolutional neural networks for bird detection in audio signals,' in *2017 25th European Signal Processing Conference (EUSIPCO)*, 2017, pp. 1764–1768. DOI: 10.23919/EUSIPCO.2017.8081512.

[83] M. Poupard, P. Best, J. Schlüter, H. Symonds, P. Spong, T. Lengagne, T. Soriano and H. Glotin, 'Large-scale unsupervised clustering of Orca vocalizations: a model for describing Orca communication systems,' in *2nd International Workshop on Vocal Interactivity in-and-between Humans, Animals and Robots*, Londres, United Kingdom, Aug. 2019. DOI: 10.7287/peerj.preprints.27979v1. [Online]. Available: https://hal.archives-ouvertes.fr/hal-02965872.

[84] J. Ford and H. Fisher, 'Group-specific dialects of killer whales (orcinus orca) in british columbia,' *Behavioral Biology of Killer Whales*, pp. 129–161, Jan. 1986.

[85] A. Cuevas, A. Veragua, S. Español-Jiménez, G. Chiang and F. Tobar, 'Unsupervised blue whale call detection using multiple time-frequency features,' in *2017 CHILEAN Conference on Electrical, Electronics Engineering, Information and Communication Technologies (CHILECON)*, 2017, pp. 1–6. DOI: 10.1109/CHILECON.2017.8229663.

[86] M. Ester, H.-P. Kriegel, J. Sander and X. Xu, 'A density-based algorithm for discovering clusters in large spatial databases with noise,' in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, ser. KDD'96, Portland, Oregon: AAAI Press, 1996, pp. 226–231.

[87] Wikipedia contributors, *Mel-frequency cepstrum — Wikipedia, the free encyclopedia*, [Online; accessed 19-June-2022], 2022. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Mel-frequency_cepstrum&oldid=1093575245.

[88] Wikipedia contributors, *Chroma feature — Wikipedia, the free encyclopedia*, [Online; accessed 19-June-2022], 2022. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Chroma_feature&oldid=1066722932.

[89] The NEMO Collaboration, L. Cosentino, M. Favetta, G. Larosa, G. Pavan, D. J. Romeo, S. Privitera and F. Speziale, *Nemo-onde: A submarine station for real-time monitoring of acoustic background installed at 2000 m depth in the mediterranean sea*, 2008. DOI: 10.48550/ARXIV.0804.2913. [Online]. Available: https://arxiv.org/abs/0804.2913.

[90] D. Gillespie, 'Detection and classification of right whale calls using an 'edge' detector operating on a smoothed spectrogram,' English, *Canadian Acoustics*, vol. 32, no. 2, pp. 39–47, Jun. 2004, ISSN: 2291-1391.

[91] M. F. Baumgartner and D. M. Fratantoni, 'Diel periodicity in both sei whale vocalization rates and the vertical migration of their copepod prey observed from ocean gliders,' *Limnology and Oceanography*, vol. 53, no. 5part2, pp. 2197–2209, 2008. DOI: https://doi.org/10.4319/lo.2008.53.5_part_2.2197. eprint: https://aslopubs.onlinelibrary.wiley.com/doi/pdf/10.4319/lo.2008.53.5_part_2.2197. [Online]. Available: https://aslopubs.onlinelibrary.wiley.com/doi/abs/10.4319/lo.2008.53.5_part_2.2197.

[92] D. K. Mellinger and C. W. Clark, 'Recognizing transient low-frequency whale sounds by spectrogram correlation,' *The Journal of the Acoustical Society of America*, vol. 107, no. 6, pp. 3518–3529, 2000. DOI: 10.1121/1.429434. eprint: https://doi.org/10.1121/1.429434. [Online]. Available: https://doi.org/10.1121/1.429434.

[93] A. Dassatti, M. van der Schaar, P. Guerrini, S. Zaugg, L. Houégnigan, A. Maguer and M. André, 'On-board underwater glider real-time acoustic environment sensing,' in *OCEANS 2011 IEEE - Spain*, 2011, pp. 1–8. DOI: 10.1109/Oceans-Spain.2011.6003482.

[94] A. Mesaros, T. Heittola and T. Virtanen, 'Tut database for acoustic scene classification and sound event detection,' in *2016 24th European Signal Processing Conference (EUSIPCO)*, 2016, pp. 1128–1132. DOI: 10.1109/EUSIPCO.2016.7760424.

[95] A. Mesaros, T. Heittola, A. Eronen and T. Virtanen, 'Acoustic event detection in real-life recordings,' Jul. 2014.

[96] T. Heittola, A. Mesaros, T. Virtanen and M. Gabbouj, 'Supervised model training for overlapping sound events based on unsupervised source separation,' in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013, pp. 8677–8681. DOI: 10.1109/ICASSP.2013.6639360.

[97] T. Heittola, A. Mesaros, A. Eronen and T. Virtanen, 'Audio context recognition using audio event histograms,' in *2010 18th European Signal Processing Conference*, 2010, pp. 1272–1276.

[98] A. Eronen, V. Peltonen, J. Tuomi, A. Klapuri, S. Fagerlund, T. Sorsa, G. Lorho and J. Huopaniemi, 'Audio-based context recognition,' *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 1, pp. 321–329, 2006. DOI: `10.1109/TSA.2005.854103`.

[99] G. Parascandolo, H. Huttunen and T. Virtanen, 'Recurrent neural networks for polyphonic sound event detection in real life recordings,' *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Mar. 2016. DOI: `10.1109/icassp.2016.7472917`. [Online]. Available: `http://dx.doi.org/10.1109/ICASSP.2016.7472917`.

[100] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior and K. Kavukcuoglu, *Wavenet: A generative model for raw audio*, 2016. arXiv: `1609.03499 [cs.SD]`.

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

[102] K. J. Piczak, 'ESC: Dataset for Environmental Sound Classification,' in *Proceedings of the 23rd Annual ACM Conference on Multimedia*, Brisbane, Australia: ACM Press, 13th Oct. 2015, pp. 1015–1018, ISBN: 978-1-4503-3459-4. DOI: `10.1145/2733373.2806390`. [Online]. Available: `http://dl.acm.org/citation.cfm?doid=2733373.2806390`.

[103] P. Warden, *Speech commands: A dataset for limited-vocabulary speech recognition*, 2018. DOI: `10.48550/ARXIV.1804.03209`. [Online]. Available: `https://arxiv.org/abs/1804.03209`.

[104] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li and L. Fei-Fei, 'Imagenet: A large-scale hierarchical image database,' in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255. DOI: `10.1109/CVPR.2009.5206848`.

[105] K. Chen, X. Du, B. Zhu, Z. Ma, T. Berg-Kirkpatrick and S. Dubnov, *Hts-at: A hierarchical token-semantic audio transformer for sound classification and detection*, 2022. DOI: `10.48550/ARXIV.2202.00874`. [Online]. Available: `https://arxiv.org/abs/2202.00874`.

[106] K. Koutini, J. Schlüter, H. Eghbal-zadeh and G. Widmer, *Efficient training of audio transformers with patchout*, 2021. DOI: `10.48550/ARXIV.2110.05069`. [Online]. Available: `https://arxiv.org/abs/2110.05069`.

[107] A. Nagrani, S. Yang, A. Arnab, A. Jansen, C. Schmid and C. Sun, *Attention bottlenecks for multimodal fusion*, 2021. DOI: 10.48550/ARXIV.2107.00135. [Online]. Available: https://arxiv.org/abs/2107.00135.

[108] A. Gazneli, G. Zimerman, T. Ridnik, G. Sharir and A. Noy, *End-to-end audio strikes back: Boosting augmentations towards an efficient audio classification network*, 2022. DOI: 10.48550/ARXIV.2204.11479. [Online]. Available: https://arxiv.org/abs/2204.11479.

[109] S. Verbitskiy, V. Berikov and V. Vyshegorodtsev, *Eranns: Efficient residual audio neural networks for audio pattern recognition*, 2021. DOI: 10.48550/ARXIV.2106.01621. [Online]. Available: https://arxiv.org/abs/2106.01621.

[110] D. Giannoulis, D. Stowell, E. Benetos, M. Rossignol, M. Lagrange and M. D. Plumbley, 'A database and challenge for acoustic scene classification and event detection,' in *21st European Signal Processing Conference (EUSIPCO 2013)*, Sep. 2013, pp. 1–5.

[111] L. Sayigh, M. Daher, J. Allen, H. Gordon, K. Joyce, C. Stuhlmann and P. Tyack, 'The watkins marine mammal sound database: An online, freely accessible resource,' vol. 27, Jan. 2016, p. 040 013. DOI: 10.1121/2.0000358.

[112] NOAA Pacific Islands Fisheries Science Center, *Pacific Islands Passive Acoustic Network (PIPAN) 10kHz Data*, 2021. DOI: 10.25921/Z787-9Y54. [Online]. Available: https://www.ncei.noaa.gov/metadata/geoportal/rest/metadata/item/gov.noaa.ncei.pad:PIFSC_HARP_10kHzDecimated/html.

[113] A. Allen, M. Harvey, L. Harrell, A. Jansen, K. Merkens, C. Wall, J. Cattiau and E. Oleson, 'A convolutional neural network for automated detection of humpback whale song in a diverse, long-term passive acoustic dataset,' *Frontiers in Marine Science*, vol. 08, p. 607 321, Mar. 2021. DOI: 10.3389/fmars.2021.607321.

[114] K. Kowarski, S. Cerchio, H. Whitehead and H. Moors-Murphy, 'Where, when, and why do western north atlantic humpback whales begin to sing?' *Bioacoustics*, vol. 31, no. 4, pp. 450–469, 2022. DOI: 10.1080/09524622.2021.1972838. eprint: https://doi.org/10.1080/09524622.2021.1972838. [Online]. Available: https://doi.org/10.1080/09524622.2021.1972838.

[115] Discovery of Sound in the Sea (DOSITS), *Seismic airguns*, https://dosits.org/animals/effects-of-sound/anthropogenic-sources/seismic-airguns/, [Online; accessed 21. June 2022].

[116] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai and S. Chintala, 'Pytorch: An imperative style, high-performance deep learning library,' in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox and R. Garnett, Eds., Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: `http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf`.

[117] Y. Gong, Y.-A. Chung and J. Glass, *AST: Audio Spectrogram Transformer*, `https://github.com/YuanGongND/ast`, Commit: 87a8004, 2022.

[118] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk and Q. V. Le, 'Specaugment: A simple data augmentation method for automatic speech recognition,' *Interspeech 2019*, Sep. 2019. DOI: `10.21437/interspeech.2019-2680`. [Online]. Available: `http://dx.doi.org/10.21437/Interspeech.2019-2680`.

[119] Y.-Y. Yang, M. Hira, Z. Ni, A. Chourdia, A. Astafurov, C. Chen, C.-F. Yeh, C. Puhrsch, D. Pollack, D. Genzel, D. Greenberg, E. Z. Yang, J. Lian, J. Mahadeokar, J. Hwang, J. Chen, P. Goldsborough, P. Roy, S. Narenthiran, S. Watanabe, S. Chintala, V. Quenneville-Bélair and Y. Shi, 'Torchaudio: Building blocks for audio and speech processing,' *arXiv preprint arXiv:2110.15018*, 2021.

[120] Z. Caceres, *Specaugment with pytorch*, `https://github.com/zcaceres/spec_augment`, Commit: 48cc7a2ed7d5ae174e54bff1436ee049eaeb2db5, 2019.

[121] Nicki Skafte Detlefsen, Jiri Borovec, Justus Schock, Ananya Harsh, Teddy Koker, Luca Di Liello, Daniel Stancl, Changsheng Quan, Maxim Grechkin and William Falcon, *TorchMetrics - Measuring Reproducibility in PyTorch*, Feb. 2022. DOI: `10.21105/joss.04101`. [Online]. Available: `https://github.com/PyTorchLightning/metrics`.

[122] W. Falcon and The PyTorch Lightning team, *PyTorch Lightning*, version 1.4, Mar. 2019. DOI: `10.5281/zenodo.3828935`. [Online]. Available: `https://github.com/PyTorchLightning/pytorch-lightning`.

[123] M. Själander, M. Jahre, G. Tufte and N. Reissmann, *EPIC: An energy-efficient, high-performance GPGPU computing research infrastructure*, 2019. arXiv: `1912.05848 [cs.DC]`.

[124] L. Biewald, *Experiment tracking with weights and biases*, Software available from wandb.com, 2020. [Online]. Available: `https://www.wandb.com/`.

[125] Wikipedia contributors, *Receiver operating characteristic — Wikipedia, the free encyclopedia*, https://en.wikipedia.org/w/index.php?title=Receiver_operating_characteristic&oldid=1094148194, [Online; accessed 29-June-2022], 2022.

[126] S. Reed, K. Zolna, E. Parisotto, S. G. Colmenarejo, A. Novikov, G. Barth-Maron, M. Gimenez, Y. Sulsky, J. Kay, J. T. Springenberg, T. Eccles, J. Bruce, A. Razavi, A. Edwards, N. Heess, Y. Chen, R. Hadsell, O. Vinyals, M. Bordbar and N. de Freitas, *A generalist agent*, 2022. DOI: 10.48550/ARXIV.2205.06175. [Online]. Available: https://arxiv.org/abs/2205.06175.

[127] N. Shvetsova, B. Chen, A. Rouditchenko, S. Thomas, B. Kingsbury, R. Feris, D. Harwath, J. Glass and H. Kuehne, *Everything at once – multi-modal fusion transformer for video retrieval*, 2021. DOI: 10.48550/ARXIV.2112.04446. [Online]. Available: https://arxiv.org/abs/2112.04446.

[128] G. Goh, N. Cammarata, C. Voss, S. Carter, M. Petrov, L. Schubert, A. Radford and C. Olah, 'Multimodal neurons in artificial neural networks,' *Distill*, 2021, https://distill.pub/2021/multimodal-neurons. DOI: 10.23915/distill.00030.

[129] Wikipedia contributors, *Cepstrum — Wikipedia, the free encyclopedia*, https://en.wikipedia.org/w/index.php?title=Cepstrum&oldid=1095367827, [Online; accessed 30-June-2022], 2022.

[130] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto and H. Adam, *Mobilenets: Efficient convolutional neural networks for mobile vision applications*, 2017. DOI: 10.48550/ARXIV.1704.04861. [Online]. Available: https://arxiv.org/abs/1704.04861.

[131] Wikipedia contributors, *Speed of sound — Wikipedia, the free encyclopedia*, https://en.wikipedia.org/w/index.php?title=Speed_of_sound&oldid=1092245868, [Online; accessed 21-June-2022], 2022.

[132] K. V. Mackenzie, 'Discussion of sea water sound-speed determinations,' *The Journal of the Acoustical Society of America*, vol. 70, no. 3, pp. 801–806, 1981. DOI: 10.1121/1.386919. eprint: https://doi.org/10.1121/1.386919. [Online]. Available: https://doi.org/10.1121/1.386919.

Martin Moan

Noise in the Sea

NTNU
Norwegian University of
Science and Technology

Akvaplan niva