Detecting Marine Acoustic Profiles With Deep-Learning Denoising

Daniel Levin
Statistics
dllevin@calpoly.edu

Jason Mulson
Statistics
jmulson@calpoly.edu

Nick Gammal
Marine Science
ngammal@calpoly.edu

01. ABSTRACT

The goal of this pipeline is to learn the soundscape of background noise in the ocean, then use it to identify anomalous audio in the data that we feed it. It takes mono audio from an omnidirectional hydrophone and turns it into a set of bounding boxes representing the duration and frequency range of marine animal calls. Our pipeline differs from most object detection neural networks in that it is mostly unsupervised. Also, a majority of the pipeline focuses on denoising the audio, whereas typical object detection focuses on finding targets accurately. Despite these unorthodox modeling techniques, we achieve excellent metrics, including 0.8223 precision. This indicates that our model knows when it found something useful. Although precision is high, the low recall (0.5140) informs us that the pipeline is not ready to be used in research settings.

02. INTRODUCTION

Motivation

Researchers use underwater audio data for studying a wide variety of topics. Marine acoustic profiles are important sources of data when studying the behavioral ecology of marine mammals such as whales, dolphins, and pinnipeds. We can apply analyses of calls they make to map their sounds with migration patterns, feeding, reproduction, and other known behaviors (NOAA). Students on the Marine Acoustics Research Team led by Maddie Schroth-Glanz at Cal Poly spent many hours listening to audio files and detecting and classifying animal sounds. Manually sifting through audio files takes lots of time, so our machine learning pipeline aims to reduce the time it takes them to comb through data. One barrier that previous attempts to build a pipeline encountered are the random noise produced in the ocean that does not belong to marine animals. To combat the messy audio, we focused on noise-reduction techniques that enhance sounds in the foreground before identifying them.

Dataset

The current dataset consists of seventy-seven raw audio files (WAV format) and thirty annotation files (text format). Each annotation file corresponds to one of the WAV files and includes lower and upper bounds for the time and frequency ranges within the audio where calls occur, as well as what species made each call. All of this data is stored within an Amazon Web Services (AWS) account belonging to Dr. Schroth-Glanz.

For our purposes, there are two noteworthy types of WAV file to work with: background noise without calls, and files including calls. The backbone of our deep-learning denoising model is the capability for decoding and re-encoding marine background signal. Because of this, we need to ensure that we have a reasonably-sized file to teach the model what not to look for. In the case of our model, we used only a 30-minute clip of audio which we were sure contained no anomalies of interest.

03. METHODS

Feature Engineering

WAV files store sampling method metadata as well as the audio itself. This data is uncompressed and the sampling rate is extremely high. High sampling rates make the data very dense, which contributes to the massive size of our WAV files. To deal with the immense file size, we decimated the WAV files before use. Decimation involves sampling at a lower rate than the specified sample rate in the raw WAV file. In our cases, we took every tenth measurement, reducing the size of our files by 90% while maintaining the integrity of all target acoustic profiles.

After decimation, we apply a short-time fourier transform (STFT) on each decimated WAV file. STFT is a signal processing technique that analyzes the frequency content of a signal over time. The most common method of visualizing STFT output uses a spectrogram, which indicates amplitude across time and frequency. Since our audio comes from the ocean and is messy, the post-STFT audio is also messy. We clean the transformed audio by implementing per-channel energy normalization.

Per-channel energy normalization (PCEN) is a normalization technique that aims to balance the volume across each frequency band. PCEN is useful because underwater animal calls vary in intensity across varying and wide frequency ranges. This variability creates bias towards specific frequencies with higher or lower amplitude intensities. So, PCEN reduces the bias by balancing the amplitude intensity to make the animal calls distinguishable.

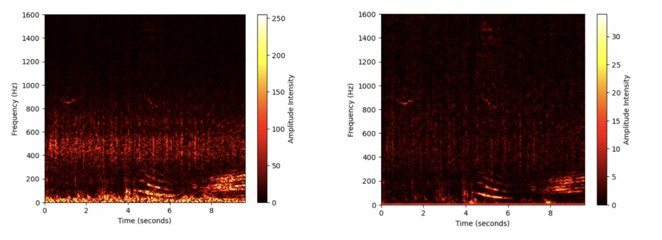


Figure 1: Pre-PCEN vs. Post-PCEN

Figure 1 shows two spectrograms before and after applying PCEN to demonstrate the effectiveness of PCEN at reducing background noise and enhancing foreground signals. The main goal of this technique is to eliminate the high-intensity noise between 0 HZ and 30 HZ. Four parameters within the PCEN function - time constant, gain, bias, and power, are the main controls we have to target specific sounds. We changed these values to work well with underwater applications.

Examining the four PCEN parameters further, time constant refers to the speed of the amplitude modulations of the target sounds; gain applies to the skewness of the background magnitudes; bias relates to the volume of the background noise; and power is based on the distance from the target. Typical use cases for PCEN include indoor audio for humans, so we had to adjust the four parameters drastically. Time constant increased from 0.4s to 4s to account for dealing with slower amplitude modulations underwater compared to indoors. Gain decreased from .98 to .7 because of a larger skew in the background noise. Bias increased heavily from 2 to 20 to account for how loud the background is. Power decreased from .5 to .2 because of the long distance from targets.

Deep-Learning Denoising

With the goal of background noise reduction, we trained a dual-paired neural network known as a Variational Autoencoder (VAE) to compress and reconstruct audio data.

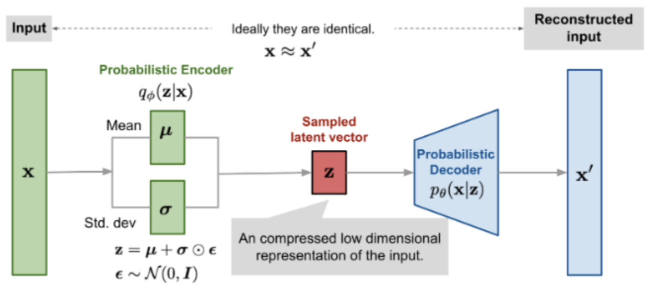

Input
Ideally they are identical.
$\mathbf{x} \approx \mathbf{x}'$
Reconstructed input
Probabilistic Encoder
$q_\phi(\mathbf{z}|\mathbf{x})$
Mean
$\boldsymbol{\mu}$
Std. dev
$\boldsymbol{\sigma}$
$\mathbf{x}$
Sampled latent vector
$\mathbf{z}$
Probabilistic Decoder
$p_\theta(\mathbf{x}|\mathbf{z})$
$\mathbf{x}'$
$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}$
$\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$
An compressed low dimensional representation of the input.

Figure 2: Variational Autoencoder

As shown in *Figure 2*, the process begins with an encoder. This segment of the pipeline performs a dimensionality reduction down to a single-dimension latent space, where additional noise is added to assist in learning the underlying structure of the dataset. From there, the data is fed into a decoder, which reconstructs the latent space audio into its original form. Model performance is evaluated based on a Mean Absolute Error (MAE) reconstruction loss where the absolute value of the input and output data point is calculated. The objective is to minimize MAE so reconstructed audio is as close to the original as possible.

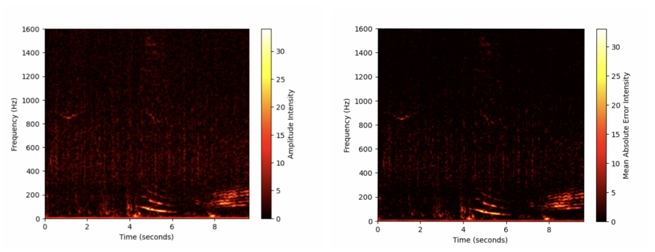


Figure 3: Deep Learning Denoising reduces fine noise

The training data for this VAE does not contain any target calls. This training scheme ensures the neural network becomes familiar with the structure of the background noise only. When the model encounters a biotic call after training on background noise, a higher MAE loss occurs compared to the soundscape around it. Heat maps of the MAE loss at each time and frequency display clean representations of the original audio as shown in *Figure 3*. In essence, this approach achieves denoising by applying deep learning to recognize noise and rebuild it correctly, while not recognizing target sounds and rebuilding them incorrectly.

Box Creation / Object Detection

With the majority of the background noise removed, we can generate bounding boxes of the remaining anomalous audio signals using open-source libraries. Scikit-maad was the library of choice for generating bounding boxes due to it being explicitly designed for bioacoustic analysis. Although Scikit-maad contains its own loading and denoising methods, we only used the pre-trained boxing method, which identifies regions of interest within soundscapes (Scikit-maad) as shown in *Figure 4*. Scikit-maad was originally designed for environmental audio, so we trust its use without alteration in our pipeline.

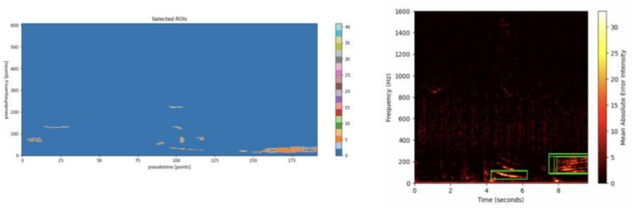


Figure 4: Generating Bounding Boxes

Post-Processing

The model evaluation script reads annotations and model output as dictionaries full of boxes. This way, each box is guaranteed to belong to a single, unique key. Each box itself has five main measurements: ID, left and right bounds (which represent start and end times), and top and bottom bounds (which represent the frequency range spanned by the box). Other box measurements important for post-processing are box duration (time length), confidence (model certainty), and two adjacency lists (one for annotations and one for other predictions). Using these sets of boxes, the model evaluation script is designed to clean up the model output from both research-oriented and objective perspectives.

Our team developed two methods that act as cleanup tools for the box predictions. The first is a widely used algorithm in object detection pipelines, called Non-Maximum Suppression (NMS). The second is a filtering method, which allows people to utilize the pipeline to filter output based on their domain knowledge. In combination, these tools are designed to select the best boxes in the eyes of both the model and the model users.

The "model perspective" cleanup is handled by Non-Maximum Suppression. NMS is a relatively simple algorithm that compares the confidence and overlap of adjacent boxes to decide which is worth keeping (for visual see "NMS" in appendix). For any pair of predicted boxes with an overlap exceeding a given threshold, only the box with highest confidence is kept. In the context of model performance, this algorithm is crucial for minimizing duplicate predictions of signals.

The “domain knowledge”-oriented cleanup is handled by filters. The filtering tool is designed so that a researcher can input a box measurement, threshold, and direction and remove boxes that don’t fit those criteria (for visual see “Filtering” in appendix). Our team found the greatest success with filters on three measurements: confidence, duration, and top (maximum frequency). The confidence threshold settled at 0.3, meaning any boxes with lower confidence than 0.3 were ignored. The duration threshold was useful in removing boxes shorter than 0.25 seconds in length, which tended to capture vertical bands in the background audio. This threshold was chosen to be about half the length of the shortest annotated call, which lasted 0.49 seconds long. The final filter removed any boxes that reached above 2400 hz. Again, this filter was chosen based on the annotations, as it is about 1.5 times the maximum annotated frequency.

Overlap Calculations

Over the course of this project, two overlap calculations were studied: Intersection Over Union (IOU) and the Overlap Coefficient. Of the two, IOU is more widely known and used. It is the ratio of the space shared by two boxes to the total space belonging to at least one of the boxes. The overlap coefficient is very similar, but it is more favorable towards boxes that are small. This calculation is the ratio between the shared space of the boxes against the area of the smaller box. For the purposes of our analysis, we used the overlap coefficient, although implementations using IOU are also included in the post-processing code.

04. RESULTS

Our pipeline performed exceptionally well in terms of precision. It knows when it found something correctly. As reflected in Table 1, our accuracy reached nearly 70%, while precision was consistently above 80% after applying NMS and filters to box predictions. Unfortunately, recall remained low.

Metric	Previous Team	Our Team
Accuracy	N/A	0.6983
Precision	0.458	0.8223
Recall	0.817	0.5140
F1	0.587	0.6326

Table 1: Model metrics with comparison to previous work

One reason for our low recall is that we output few predictions relative to the ground truth call count in the test set. The metrics in Table 1 reflect an output of 1731 boxes on a file containing 2350 true targets.

05. DISCUSSION

Results in a Research Context

Our pipeline does not perform well enough to be used in a research setting as a reliable detector of marine mammal calls. Although it boasts decent accuracy and exceptional precision, the low recall shows that it misses too many calls. Based on visual and auditory exploration of the denoising efforts, it is clear that our PCEN pre-processing and deep-learning denoising are exceptionally good at cleaning the audio files. When played side-by-side, there is a clear improvement in the clarity of sounds belonging to targets. Where our models tend to struggle are faint calls or calls that exist in particularly noisy regions of the soundscape.

Comparison Against Previous Work

Previous research efforts focused entirely on creating the best boxes possible. PCEN was attempted, but not successfully implemented in an easily replicable way. Convolutional neural networks were used to create boxes from spectrograms.

Our improvements to the PCEN methodology and implementation of the deep-learning denoising model produced significantly more clear audio to work with. Previous work used significantly more data to train on, and resulted in excessively high call prediction counts. For example, the best previous model predicting on our test set (with 2350 true targets) produced over 6000 boxes. Recall was high, but it was inflated because there were far more boxes than would be reasonable to predict, and the other metrics suffered as a result. Our model, which produces much lower prediction counts finds targets correctly, with a very low false positive rate (0.1507). Ideally, we need to make some improvements to increase our box count while maintaining high precision and accuracy.

06. FUTURE WORK

Improvements to Existing Pipeline

Looking at the big picture, we need to optimize the denoising at both pre-processing and modeling stages so that it continues to remove background noise without diminishing or otherwise worsening the quality of target calls. This will involve optimizing PCEN parameters, training the VAE on different sets of background noise, and tuning the sensitivity of the VAE to better reconstruct faint calls.

An improvement that can be made to feature engineering involves optimizing the parameters within the PCEN function (time constant, gain, bias, and power). Our pipeline includes no method of determining the best values for the parameters other than testing many different combinations manually and looking at the resulting spectrograms. More in-depth analysis of these parameters and their qualitative effects on the transformed audio should be conducted.

Necessary improvements to the VAE include changing the training protocol and hyperparameter tuning. The current training system only uses about 30 minutes of background noise. This is potentially unreliable because the dataset spans multiple months, but is only trained on one, continuous 30 minute segment. We have introduced heavy bias towards the date of this particular audio clip. As for hyperparameter tuning, it is unclear whether the VAE is responsible for diminishing faint calls, but changes to its parameters would be worth exploring.

Lastly, there is an unfinished method for box combination that is an alternative to NMS. The original intention behind this cleanup tool was to capture harmonics, which are characteristic of many whale calls and songs. The idea is that we could combine boxes which overlap in the time dimension, thereby covering the acoustic profile at each harmonic in one box as opposed to separately. With the current pipeline implementation, it simply does not work.

Signal Classification

A major goal of this project in the long run is classification of calls. Our efforts focused entirely on finding the calls without classification. In order for this pipeline to be useful in a research context, we need to be able to accurately predict not only where in the audio a call exists, but also what organism each call belongs to. This can be added as a new model, implemented after box creation in the pipeline, or it could be applied along with the boxing method in a unified model.

07. CONCLUSION

In conclusion, we have successfully implemented an audio denoising and object detection pipeline using supervised and unsupervised learning. The model performs reasonably well, producing high-precision, accurate predictions of marine mammal calls. It lacks sensitivity to faint calls and calls that occur in regions that are exceptionally noisy. Future work should focus on better training and tuning protocols, improved pre- and post-processing methods, and a new classification implementation. These efforts will lead to improved recall while hopefully maintaining high precision, and classification will ensure greater utility to future research.

08. ACKNOWLEDGEMENTS

Maddie Schroth-Glanz provided invaluable support throughout this project. Her insights and dedication were instrumental in shaping the success of our project. We also want to thank Alison Stimpert of Moss Landing Marine Laboratories for providing the data, and Cal Poly student researchers Adelle Wilkin and Sophie Short for annotating it. Without them, our project could not have been as successful as it is. Lastly, we appreciate our professors, Dr. Alex Dekhtyar and Dr. Hunter Glanz, for their guidance throughout this project.

09. REFERENCES

Fisheries, NOAA. “Marine Mammal Acoustic Projects.” *NOAA*, 27 Apr. 2023, www.fisheries.noaa.gov/new-england-mid-atlantic/science-data/marine-mammal-acoustic-projects.

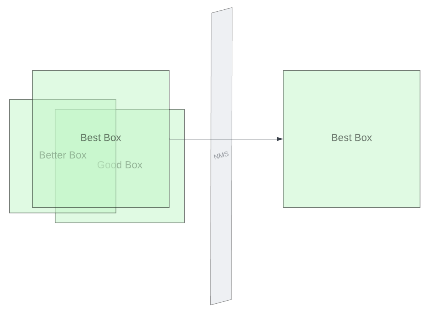
Lostanlen, Vincent. *Per Channel Energy Normalization: Why and How*, Nov. 2018, www.justinsalamon.com/uploads/4/3/9/4/4394963/lostanlen_pcen_spl2018.pdf.

Scikit-Maad. “Scikit Maad (Soundscape Analysis).” *GitHub*, 15 Nov. 2022, github.com/scikit-maad/scikit-maad.

“The Short-Time Fourier Transform: Spectral Audio Signal Processing.” *DSP*, www.dsprelated.com/freebooks/sasp/Short_Time_Fourier_Transform.html. Accessed 14 June 2023.

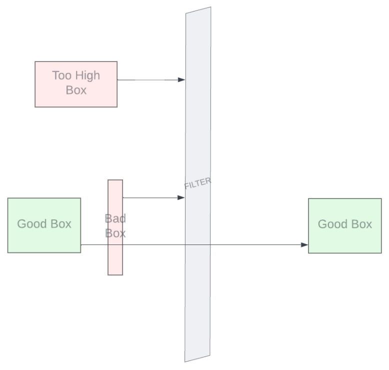
10. APPENDIX

NMS: Non-Maximum Suppression



Non maximum suppression selects the box with maximum confidence out of a set of boxes that exceed a threshold for acceptable overlap between predictions. For example, the visual above shows three boxes with high overlap. There is a "good", "better", and "best" box, and only the best box is retained by the NMS algorithm.

Filtering:



Filtering takes a set of boxes and returns only the boxes that satisfy a given condition. For example, the visual above shows three boxes passing into a filter. The top box is too high, and the rightmost box is too short, so they don't pass through the filter.