M3.2 NLP

# Proposal

2024-07-09

Jaeden Amero

## Introduction

*Project topic*

Build a system to send a notification when orca sounds are detected in a real-time continuous audio-stream.

*Domain of project*

Audio signal processing. Data comes from hydrophones located throughout the Salish Sea, provided by the Orcasound team (orcasound.net). Orcasound provides the data for "citizen science" and public outreach. The data continues to raise awareness about orcas and their natural habitat and how humans are disturbing their habitat.

*Motivation*

The hydrophone audio feed is usually very boring, containing water slapping sounds or a droning mechanical hum. A notification system would alert the public when they could hear an orca on the livestream. Tracking when orca sounds are heard passing by a certain hydrophone site would indicate orca migratory habits and where an orca pod may currently be located.

## Data Description

Type: Audio

Source: orcasound.net, an online dataset

Key attributes:

- 7 years of nearly uninterrupted hydrophone data
- 7 hydrophone sites
- 48 kHz audio sampling rate
- Stereo hydrophone data available for a subset of site locations
- Pure unlabeled audio
- New data livestreamed continually

## Data Preparation

**From M3.1**

*For training:*

Unbalanced dataset. Mostly boring. Not that often we have an interesting biosignature.

Data processing: Sample-rate conversion, Channel conversion (to mono),

Normalization (consistent volume)

Noise reduction, silence removal -- not doing

Data augmentation, like pitch shifting or speed alteration, or adding noise-- not

doing. The idea would be to increase the variability and robustness of the dataset, but our hypothesis is that we have enough data already.

Segmentation: Chunking/slicing -- yes. We'll split long audio files into shorter chunks

No annotation for training

Use automated tools to check for issues like clipping (We'll use sox, an open-source command line program useful for working with audio files)

*For testing:*

Use a labeled dataset to check that our model can identify when an orca is present within an audio recording chunk

*For inference:*

Any data preparation done for inference must be performed in real-time, so we will limit the processing only to resampling the data for compatibility with existing pretrained audio models.

Proper data preparation will let us re-use the pretrained AST model. The model requires a certain sample rate and audio length, so resampling and slicing/chunking is essential. Normalization is also important because it prevents the model from being influenced by varying volume levels (which can lead to inconsistent performance).

## Next Steps & Overall Goals

**New for M3.2**

With data preparation complete, we use our dataset to finetune the Audio Spectrogram Transformer (AST). We reserve a portion of our training set for use as a test set.

First, we extend the AST, which is decoder only, with an encoder. This enables us to fine tune the model without a labeled dataset. We follow an unsupervised learning approach, training the AST auto-encoder to learn a lower dimensional representation of our dataset.

The resulting decoder portion of our model is used to create an embedding for a user-provided audio file query. For example, a short snippet of an orca call.

Then, we use a vector database to search for a previously heard sound that matches the audio query, using the generated embedding for the query.

By querying the model a few times with known orca audio samples, we learn where orca calls are clustered within the model's latent space.

When the system operates, the vector database is queried continually with new audio queries in real-time. If the cosine distance within the latent space is close enough to other vectors we've identified as orcas, we'll consider that we found an orca.

Once we've identified an orca, we publish the event to a database that supports publish/subscribe semantics.