Harsha Jagarlamudi, Kelly Kong

DSC 180B: A09

Mentor: Ryan Kastner, Jacob Ayers

9 March 2022

Cross-Correlation Template Based Matching

1. Introduction

Climate change is a global forefront issue that affects all species. Tracking the biodiversity of species and their population can help gauge how much climate change has affected these species. Within our domain, the focus is on birds, so we use bird audio to measure their biodiversity and to potentially monitor endangered birds. The intent behind this project is that by being able to passively identify and differentiate bird species in a reliable way, it will be possible to gain a better understanding of the environments they belong to. We can determine what bird species belong to what habitats and get a stronger grasp of changes to an ecosystem based on the species we believe to be present there.

However, the problem is that a lot of machine learning models specifically for acoustic species IDing need labeled data that require manual input and time. Within our section, we saw this as we all had to manually label over 200 bird clips. This process can be very time-consuming, can be prone to inconsistencies between different human annotators, and can have human input errors that would be less expected of an automated solution.

2. Methods

The cross-correlation based template-matching project involves Digital Signal Processing for audio data segmentation. This technique takes a sample of audio and compares it to a set of

other audio, outputting a score array based on similarity to the sample. Then, an algorithm is run to determine the best scoring audio clips which are hopefully similar to the sample and can be used as training data for scalable models. Our intention with this project is to create a reliable way to automatically label audio data by species and cut down the time investment and manual input needed to create strong training data for machine learning models that work on acoustic species IDing.

The cross-correlation pipeline is Python-based and uses the packages PyHa, Librosa, and SciPy. PyHa is a package designed to convert audio-based “weak” labels to strong intraclip labels. It provides a pipeline to compare automated moment-to-moment labels to human labels by combining Python and Piha.

The first step in performing cross-correlation is deciding on data to run the pipeline on. In our case, to make sure cross-correlation was working as intended initially, we chose to run the pipeline on ScreamingPiha audio data. The first function we call is load_audio, which takes in a clip-path and loads the audio data at 12kHz into a variable that is returned.

Following this, we want to do a simple exploratory data analysis on this clip and create a spectrogram to take a look at. This function, spectrogram, takes the output of load_audio as input and short-time Fourier transforms the audio. This transformed audio is saved to a variable, used to display a spectrogram of the data, and then returned as an output.

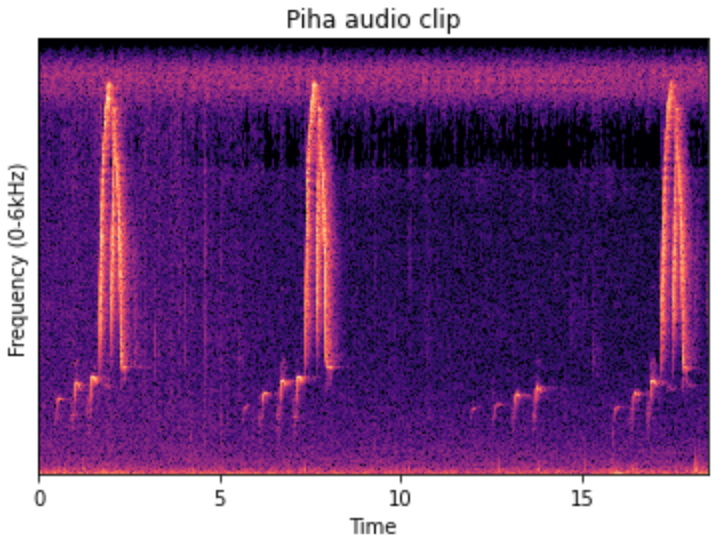

Figure 1: Screaming Piha Audio Clip Spectrogram

The audio clip spectrogram is an important piece of information to look at when performing cross-correlation on individual clips because it can provide a lot of useful information for understanding the data that is being worked with. Primarily, this spectrogram gives visual representations of what the peaks, which are usually the bird calls, look like. This should be consistent with the template and it also gives a good understanding of how the calls are spaced out in terms of time.

Next, we want to use an audio file from the same species to create an audio template. In our case, we use the function, template, that takes in the output of load_audio. A section of the audio file that contains the species sound is short-time Fourier transformed and saved as a template. This template is then plotted, displayed, and returned as the output of the function.

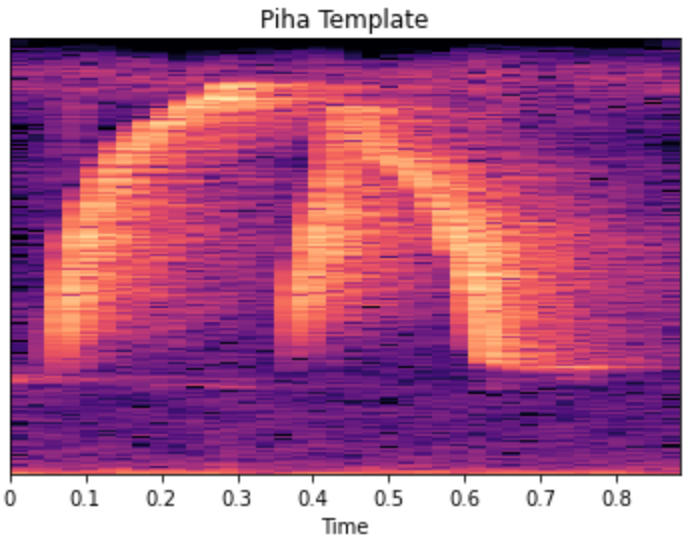

Figure 2: Screaming Piha Audio Template

This audio template can be seen used to understand what shape denotes the presence of a particular bird's call in an audio clip. While it may be more useful for human annotations, it does provide a good frame of reference for analyzing the performance of cross-correlation in cases where human annotations are not available to compare to.

Finally, we want to perform the actual cross-correlation. For this, we use the function, correlation, which takes in as input the clip-path, and the output of load_audio, spectrogram, and template. This function defines isolation parameters and uses the scipy.correlate2d function to perform correlation on the transformed audio using our template. Using PyHa's steinberg_isolate function, we created a resulting data frame and were able to display a spectrogram of peaks alongside a local score array that represented our cross-correlation model's prediction on the audio.

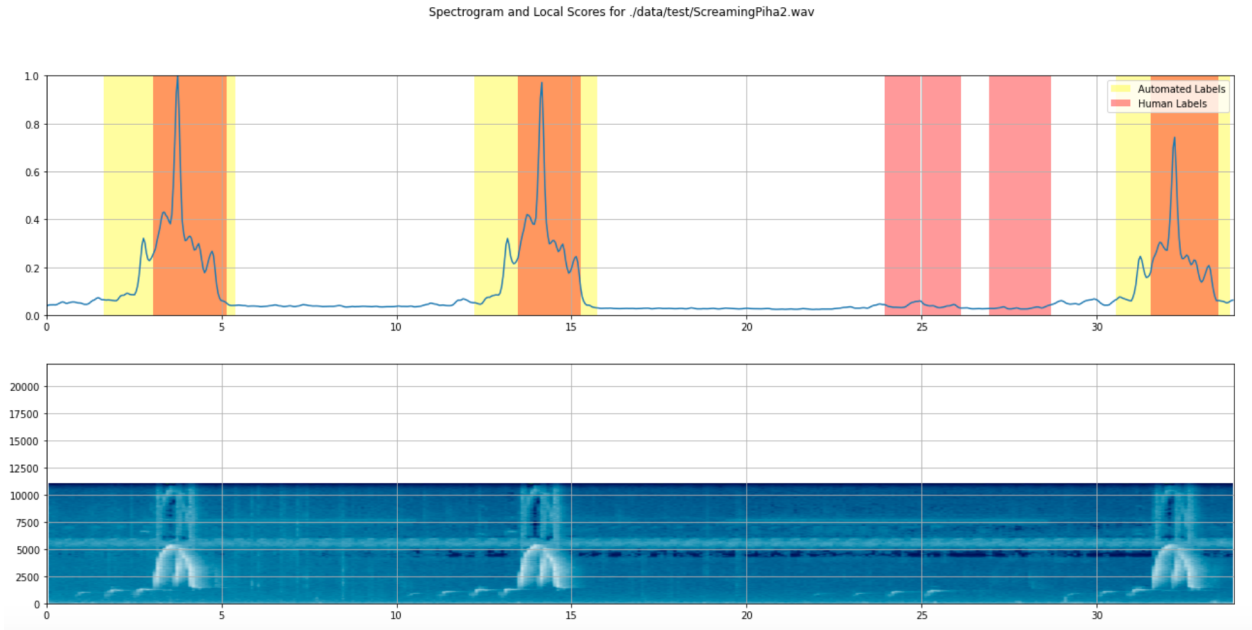

Figure 3: 2D Cross-Correlation Spectrogram and Local Score Array

Here we can see that overall, cross-correlation does a good job of annotating the main parts of the bird calls. Although the automated annotations couldn't detect the preliminary calls as a human annotator would (shown in the three pink human annotations right before the main third peak), our labeled data for the machine learning models wouldn't require it.

3. Results

In order to test the effectiveness of our current cross-correlation pipeline, we created a template from the ScreamingPiha1.wav file and used it to run cross-correlation across the other 10 ScreamingPiha WAV files. The results were as follows:

ScreamingPiha2.wav	The cross-correlation captured all of the peaks found in this file.
ScreamingPiha3.wav	The cross-correlation captured all of the peaks found in this file.
ScreamingPiha4.wav	The cross-correlation only captured the latter of the 2 peaks found in this file.
ScreamingPiha5.wav	The cross-correlation captured all 4 of the peaks found in this file.

ScreamingPiha6.wav	The cross-correlation captured 6 of the 8 peaks in this file but missed 2 of them and incorrectly labeled a section of the audio as a peak where there was none.
ScreamingPiha7.wav	The cross-correlation captured 5 of the 9 peaks in this file but the spectrogram was very noisy and this was also difficult to identify manually.
ScreamingPiha8.wav	The cross-correlation captured both peaks in this file but it also incorrectly labeled a section of the audio as a peak where there was none.
ScreamingPiha9.wav	The cross-correlation captured all 5 of the peaks found in this file.
ScreamingPiha10.wav	The cross-correlation captured all 9 of the peaks found in this file but the window on the last peak it identified was unnecessarily large.
ScreamingPiha11.wav	The cross-correlation captured all 7 of the peaks found in this file but it also incorrectly labeled a section of the audio as a peak where there was none.

Table 1: Screaming Piha Cross-Correlation Testing

In general, the cross-correlation algorithm, albeit slow, was quite effective at detecting peaks. In some cases where the spectrogram of the audio file was noisy, it could be difficult to detect some peaks and may have incorrectly deemed a section of the audio as a peak where there is none. However, in all of the files, it captured at least 50% of the peaks and in 7 of the 10 audio files we ran the template across, it captured all of the peaks successfully. This pipeline took 40 minutes to run on the 10 distinct audio files of varying lengths.

4. Conclusion

Overall, we were happy with the performance of the model at the current stage but feel that it could be improved further. It was moderately accurate at detecting peaks of the data but we want to minimize false positives and negatives to approach classification levels closer to that of human annotation. Additionally, running cross-correlation on 10 distinct audio clips took about 40 minutes

which when done by a human, would typically take closer to 10 minutes to perform. Cross-correlation does have the advantage of not requiring active input to label data but paired with its imperfect accuracy, we felt that it would currently be preferable to continue performing human annotations to get labeled data.

Moving forward, we want to improve the speed of the cross-correlation algorithm to run it more effectively on larger sets of data. In addition, we want to manually evaluate its effectiveness on different sets of data beyond the Screaming Piha, potentially with different isolation parameters. In addition, we intend to make the pipeline both more fluid and simpler to use so that any template can be passed in alongside full audio data and generate local score arrays with little manual input.

5. Individual Contributions

Both Harsha and Kelly ran and helped to clean up the initial cross-correlation notebooks provided as a starter template. Both also manually annotated various bird audio spectrograms to get a better understanding of this specific medium of data.

Harsha turned the original cross-correlation notebook into a Python file and developed functions that would be easier to call in a run script. Harsha also created the project GitHub repository alongside the file structure. Harsha had access to the Scripps Coastal Reserve audio and explored it.

Kelly created the run scripts to perform cross-correlation on the data. Kelly also created the data parameters file and incorporated it into the run script. Kelly was primarily responsible for using the cross-correlation pipeline on the UCSD Engineers for Exploration ML machine and worked to resolve errors that were encountered. Kelly had access to the Peru dataset and explored it.

6. Accomplishments and Incomplete Goals

With this project, we utilized and simplified a cross-correlation template based matching pipeline. We ran the pipeline across 10 distinct audio files for the Screaming Piha and derived conclusions about the performance of the pipeline on this bird. We were able to draw conclusions about the effectiveness of this pipeline and its viability in more specific uses given improvements.

In terms of incomplete goals, there were many different avenues we wanted to pursue but could not due to either time constraints or errors encountered. One goal we intended to pursue but abandoned due to not feeling we had enough time was improving the speed of the cross-correlation pipeline. At the moment, it can take up to 5 minutes to run the pipeline on a 30-second audio clip which makes this a very time-consuming and time inefficient process. Making the improvements to this process would involve working around the Scipy and Librosa functions used which would take much more time not only to develop but also to understand and incorporate. Another goal we could not finish was running the pipeline on large amounts of data from the Scripps Coastal Reserve and Peru datasets. We were able to take a look at this data from hard drives and process small amounts of it on the UCSD Engineers for Exploration ML machine but we frequently encountered errors when running the pipeline in batches. Coupled with the lengthy computations of the pipeline, we were only able to resolve some errors but did not feel we had sufficient time to address all of them and proceed with branching out to more species than the Screaming Piha.

7. Relevant Links

Artifact (Overview) Repository

https://github.com/kel-kong/Cross-Correlation-Artifact

Code Repository

https://github.com/hjagarla/DSC180A_A09_cross_correlation

Website

http://hjagarla.github.io/Cross_Correlation_Website

Website Repository

https://github.com/hjagarla/Cross_Correlation_Website

Peru Stratified Random Sample

https://drive.google.com/file/d/19Xk-jiCkZOiuYt2otbJsSJ5N_tidp83G/view

PyHa Reference

https://github.com/UCSD-E4E/PyHa