ASEE AMERICAN SOCIETY FOR ENGINEERING EDUCATION

Paper ID #44896

Automating Annotations for Leopard Vocalizations

Mr. Hammad Mansoor, Fairfield University

Hammad Mansoor is a passionate programmer currently pursuing a Master's degree in Software Engineering at Fairfield University. He is a graduate assistant at the Fredrickson Family Innovation Lab. With a strong affinity for Python, he enjoys crafting elegant code and has gained valuable experience in applied machine learning. Outside of the digital world, he finds solace in the great outdoors and often explores nature through hiking.

Danushka Bandara, Fairfield University

Danushka Bandara received the bachelor's degree in Electrical Engineering from the University of Moratuwa, Sri Lanka, in 2009. He received his master's and Ph.D. degrees in Computer Engineering and Electrical and Computer Engineering from Syracuse University, Syracuse, NY, USA, in 2013 and 2018, respectively. From 2019 to 2020, he worked as a Data Scientist at Corning Incorporated, Corning, NY, USA. Currently, he is an Assistant Professor of Computer Science and Engineering at Fairfield University, Fairfield, CT, USA. His Current research interests include Applied machine learning, Bioinformatics, Human-computer interaction, and Computational social science.

Automating Detection and Tracking of Leopard Vocalizations

Abstract

Animal vocalizations provide a wealth of information on animals and their surrounding environment. This acoustic data can help us understand the behavioral, physical, and mental state of an animal, which can further help biologists better support the animal's health and well-being. Our project aims to create an automated process using which biologists can identify and annotate Leopard vocalizations from recordings made at animal enclosures. These annotations will be used to study correlations between vocal characteristics and estrus in leopards. Currently, each 24-hour recording takes upwards of 74 minutes to annotate, and each clip has to be manually extracted from the main recording. This is a manual task that requires an individual to carefully go through the recording and verify instances of leopard calls. Our focus is, therefore, not only to reduce this annotation time but also to better allocate resources to more pressing tasks. For this, we use 'auditok', an Audio Activity Detection tool to isolate sounds that are above a certain energy threshold and then store them as individual clips while also retaining timestamp information. Preliminary tests on denoised recordings reveal that most animal vocalizations last about 30 seconds and occur in groups of 15 - 20 individual calls. Using 'auditok' and a Python pipeline, we can successfully isolate these individual calls and store them as clips. We also store start and end times for each vocalization in an output file for reference. Currently, we are able to identify all instances of grouped calls and ~80% of individual calls within each group within 30 - 50 seconds per file.

Introduction

Monitoring and analysis of animal behavior is important for biologists and zookeepers to better support animal health and safety. This includes studying reproductive behavior, which is an important factor in the conservation and continuation of animal species. Anecdotally from the zookeepers' experience, female leopards show an increase in their frequency of vocalizations when in estrus. [1] One of these vocalizations is the "Saw Call' which is a continuous set of long and loud vocalizations that occur in bouts of >= 5 per call.

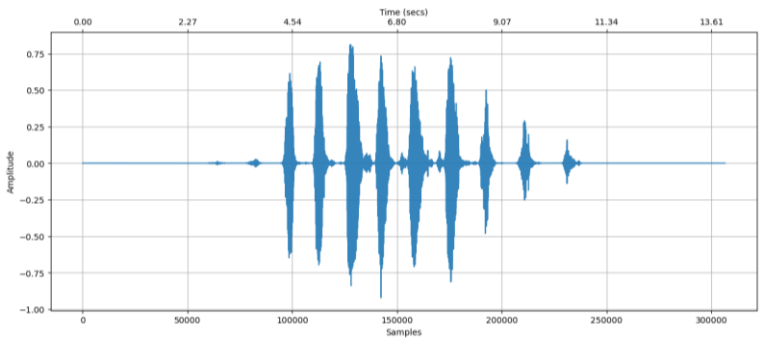

Fig. 1 Leopard saw call with 9 individual saws

Biologists studying leopards at the Connecticut Beardsley Zoo use song meters [2] from Wildlife Acoustics to collect 24-hour recordings of the sounds within the animal enclosure. Their goal is to meticulously go through these recordings and annotate all instances of vocalizations made by the leopards including the saw calls. This process of annotation requires individuals to spend hours working on these recordings. On average, it takes 74 minutes to annotate a single 24-hour recording. This is a time-consuming process but has potential for automation.

We propose a pipeline to reduce manual annotation time. The first step involves eliminating any background noise, such as static, environmental disturbance, or the sound of cars passing by. This is important as noise can lead to false positives. Once we have the denoised sound recording, we use a sound gate that marks all sounds that are above a specific energy threshold and lie within a specified time range. Since most of the noise has been removed, all that remains is the sounds within the enclosure, most of which are leopard vocalizations. Using the process mentioned, we can successfully isolate and extract these vocalizations. We save each identified vocalization as its own sound clip and add it to a running list with timestamp information and the filename within which it was found.

We also ensure that multiple sound files can be processed using this pipeline in one go. This allows us to significantly reduce manual annotation workload and verify and compare the results that we receive.

Literature review

Oswald et. al. [3] described the use of an energy threshold of an incoming signal in the frequency domain to check whether a specific sound is present in a recording. This is the focus of our approach to the detection of leopard vocalizations as this is much simpler compared to training an entire model. Astuti et. al. proposed the use of unusual animal activity detection to predict oncoming disasters and allowing governments to take necessary steps to protect its citizens. They used a SVM (support vector machine) based model to train and test the detection of animal sounds. The focus of this paper was also on birds as they are most seen in high traffic areas. [4] They used MFCC features extracted from digital animal speech signals to train said model. Zachary et. al. used a deep convolutional neural network with six trainable layers and trained it on a dataset of spectrograms generated from animal sounds. Their dataset had 17 target classes which were made up of short sound clips produced by 14 species of birds and mammals. Their model performed especially well for most owl species, with each species having an ROC-AUC value of 0.97 – 0.99. [5]
Neha et. al. used a combination of machine learning and deep learning models to classify sounds made by animals. They used mel spectrograms for each sound clip to train a convolutional neural network. They also used the extracted features from CNN to train an SVM and XGBoost model. Both had an accuracy of 68% and 72% respectively, while the CNN model had an accuracy of 98.95%. [6]

Nunes et. al. used a wearable device worn by horses to extract and store sounds like chewing or biting. They proposed the use of a RNN with a long short-term memory layer to distinguish between different noise events. They trained the model on the sounds that they received from the wearable devices and were able to achieve an accuracy of 88.64% for bites and 94.13% for chewing. Their focus was to use this model and its predictions to study foraging behaviors in horses. This study is very similar to ours in the sense that it aims to use sounds produced by animals to study their relationship to their environment [7]. Ines et. al. proposed using a new approach, few-shot learning for detection and classification of animal sounds. They realized that gathering and labelling a suitable sized dataset of animal vocalizations was a massive undertaking. It involved a significant amount of money and time being invested in it. Adapting a few-shot learning approach would allow the system to detect vocalizations even when provided with as few as 5 examples. They tested the performance of such a model on various open datasets and saw that datasets with high time resolution performed better. [8]

Methodology

The first step in creating a pipeline for annotation is to study the characteristics of the sound. From preliminary analysis and information obtained from the team at the zoo, we know that these recordings are 24 hours long and are noisy. This noise is mostly static and ambient sounds. There are a few instances where we can hear people talking or cars passing by, but since these are not as prevalent, we choose to ignore them for this step. Another prevalent noise is a banging sound. This is created when the leopard in the enclosure drops or hits something. This is common and shows no discernable pattern. This clearly shows that we need to clean this data to be able to extract any meaningful information from it.

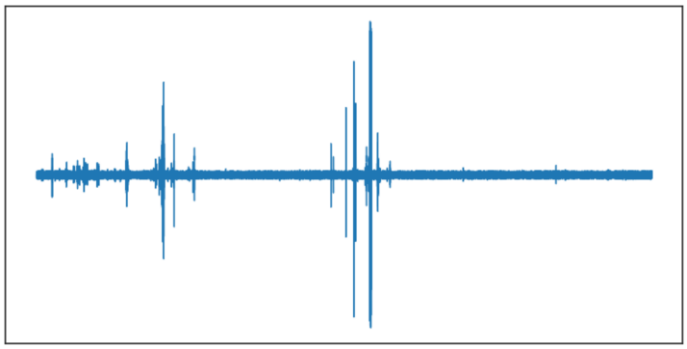

Fig. 2 Raw audio recording from zoo enclosure

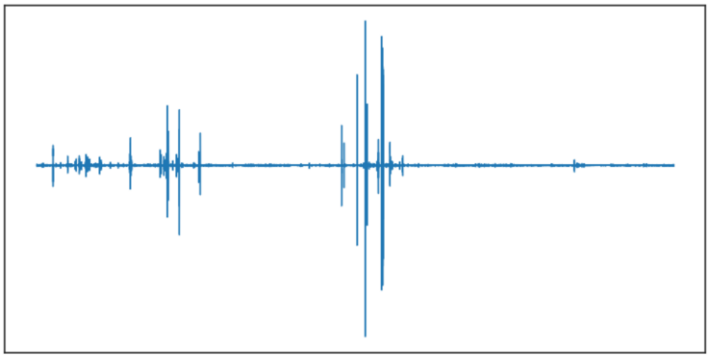

Fig. 3 Denoised version of recording

We use a spectral gating algorithm implemented in the ‘noisereduce’ Python library. It estimates the noise threshold for each frequency band of the provided signal. It uses this information to gate all noise that is below that threshold. We used a pipeline that processes an entire directory of files and stores a version that has no background noise.

Once we had a clean sound profile, we did a study of previously identified components of a saw call. We saw that on average these individual vocalizations were between 0.2 to 0.5 seconds long. The reason we focused on these instead of the entire saw call was because they were a lot more consistent based on how long they lasted. A saw call could have anywhere from 7 to 15 components meaning that they were more inconsistent in terms of a time range.

We decided to approach the problem of identifying these vocalizations in a way where we would not have to use training data. The reason was to ensure that it could be used in a ‘zero shot’ setting without the need for training a model. This enables even non-technical users like biologists to adjust the detection parameters without the need of experts in machine learning.

This direction led us to use ‘auditok,’ which is an audio tokenization tool that marks specific sound segments based on their energy. Denoising the sound helped here since the module performs better for recording without background noise. The module computes the energy of sound within a given time range. The energy is calculated as the logarithm of the RMS (root mean square) value of the sound. This calculated energy is compared to a threshold value that can be changed based on the requirement. All sound segments that have an energy greater than this threshold are detected.

Selection of a suitable threshold value involved comparing values. The ‘auditok’ module uses 55 as a default threshold value. We decided to use this as a reference value. We moved two - three energy points over if our current threshold was not suitable. Using 55 meant, we detected many extraneous sounds such as banging or the animal hitting things with its tail. The goal we realized was to ensure that these unnecessary detections were as few as possible. Moving up to 57 we saw that the number of extra detections was reduced and anticipating a similar trend, we decided to move further up. At 60, we saw that there was a clear majority of correct detections of leopard vocalizations. The extraneous vocalizations were very few and

scattered. Moving further up though, we saw that many quieter vocalizations were being missed. For example, in figure 4, we would miss the last two detections in the sequence. Due to these observations, we finalized 60 as a suitable energy threshold.

Each of these detected segments was between 0.2 to 0.5 seconds long based on the average duration of a vocalization.

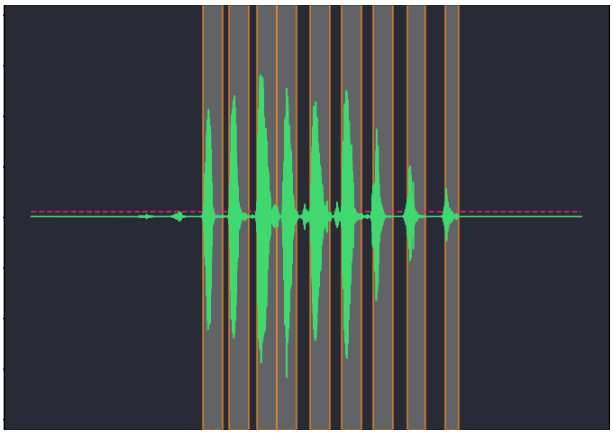

Fig. 4 Marked regions identified by ‘auditok’. Each region is stored as an individual clip for further analysis.

Our pipeline would process all sound recordings within a provided directory and save any detections made within a folder that had the same name as the recording being processed. Each sound clip corresponds to an individual vocalization and its filename has the timestamp information for its position in the original recording. Finally, after all recordings are processed, the pipeline outputs an Excel sheet where each row corresponds to a single detection called a region and also has timing information for where the sound is positioned in the recording. These detections are grouped by the files from which they were extracted. This Excel sheet is important as it helps us verify the results and test the performance and accuracy of the process.

	File	Region	Start Time	End Time
1	SMM07257_20221117_153302	1	0:27:17.95	0:27:18.45
2	SMM07257_20221117_153302	2	0:27:20.70	0:27:21.20
3	SMM07257_20221117_153302	3	0:27:21.95	0:27:22.45
4	SMM07257_20221117_153302	4	0:27:23.30	0:27:23.75
5	SMM07257_20221117_153302	5	0:27:25.60	0:27:26.05
6	SMM07257_20221117_153302	6	0:27:26.15	0:27:26.55
7	SMM07257_20221117_153302	7	0:27:26.60	0:27:26.95
8	SMM07257_20221117_153302	8	0:27:27.15	0:27:27.65
9	SMM07257_20221117_153302	9	0:27:27.65	0:27:28.15

Fig. 5 Excel output for extracted clips including start and end times.

Results and Discussion

For testing the performance of the pipeline, we decided to process 4 sound recordings that were considered diverse in their composition of different kinds of sounds. These sounds included background noise, animals drinking water, banging, things falling on the floor, and people talking. Our plan was to process these recordings and share them with the biology team we were working with and ask them to verify our results. Most of these sounds were easily eliminated using the noise removal pipeline but the banging remained. This was because the characteristics of the banging sound were very similar to that of the leopard calls. Removing them would mean losing a majority of the instances of the sawing call. We decided to let them remain and test the performance of the tokenization module.

Using an energy threshold of 60, we made a total of 490 detections from the 4 selected files. Of these, 278 were correct, and of the individual vocalizations that were part of the saw call, while 212 detections were wrong. All the wrong detections were banging sounds. This meant that the precision of the model was 0.56.

Even though a decent number of detections were incorrect, we saw that the model was able to identify all instances of a saw call that existed in these recordings. Within each of the saw calls, where each had a varying number of individual vocalizations, the model was able to identify 80.11% of them. Any vocalizations that were missed were usually those that were at the start and end of a saw call. This was because when a saw call is made, the start and end vocalizations are usually of lower intensities, and intensity increases as the call reaches its peak.

Another important point to consider is that all incorrect detections were of the banging sound. This is primarily due to the similarity of the characteristics between the individual vocalizations and the banging sound. Since our focus was to build a simpler model that identified all instances of saw calls, these banging sounds are harder to ignore as they are very similar. A more complex model that uses machine learning to distinguish between the two would be a good option as a next step.

Conclusions and Implications

Our model’s focus was to extract saw calls from 24-hour-long recordings. We were successful in building a simple pipeline that was able to identify these saw calls but also falsely extracted banging sounds. Even though this isn't ideal, it does significantly reduce the time it would take for an individual to annotate these saw calls. This is because unlike earlier, where they would have to go through the entire recording and manually listen and identify saw call groups, they can use the output of the pipeline and focus on verifying the correctness of the identifications.

This means they do not have to listen to other sounds that might not be sawing calls.

Moreover, since all saw calls occur in groups, they can easily conclude that contiguous detections are sets of saw calls. This means they can also identify missed vocalizations that exist at the start and end of each call.

These improvements in detection will significantly reduce the amount of annotation time for each recording as the human would only engage in the verification of results. Further, these results can allow us to quickly build datasets for these saw calls that can be used as training data for machine learning models. Future research could focus on using these datasets to train complex ML models that would be able to distinguish between the banging sounds and saw calls.

Bibliography

[1] https://www.wildlifeacoustics.com/products/song-meter-sm4

[2] Allwin, B., Kalignan, P. A., Nag, P. B. S., Gopikrishnan, D., & Gokarn, N. S. (2016). The reproductive behavior of Indian leopards (Panthera pardus fusca). *Journal of Veterinary Science and Technology*, *7*(5), 358.

[3] Oswald, J. N., Erbe, C., Gannon, W. L., Madhusudhana, S., & Thomas, J. A. (2022). Detection and classification methods for animal sounds. *Exploring Animal Behavior Through Sound*, *1*, 269-317.

[4] Astuti, W., Aibinu, A. M., Salami, M. E., Akmelawati, R., & Muthalif, A. G. (2011, May). Animal sound activity detection using multi-class support vector machines. In *2011 4th international conference on mechatronics (ICOM)* (pp. 1-5). IEEE.

[5] Ruff, Z. J., Lesmeister, D. B., Appel, C. L., & Sullivan, C. M. (2021). Workflow and convolutional neural network for automated identification of animal sounds. *Ecological Indicators*, *124*, 107419.

[6] Singh, N. (2020). Classification of animal sound using convolutional neural network.

[7] Nunes, L., Ampatzidis, Y., Costa, L., & Wallau, M. (2021). Horse foraging behavior detection using sound recognition techniques and artificial intelligence. *Computers and Electronics in Agriculture*, *183*, 106080.

[8] Nolasco, I., Singh, S., Morfi, V., Lostanlen, V., Strandburg-Peshkin, A., Vidaña-Vila, E., ... & Stowell, D. (2023). Learning to detect an animal sound from five examples. Ecological informatics, 77, 102258.