Automatic Passive Acoustic Monitoring for Wildlife Conservation: Lightweight Trigger-Call and Bidirectional Segmentation

Paper id: 219

Abstract

We improve the state of the art for detecting elephant calls from acoustic data on two different tasks of interest to wildlife conservation: (1) A segmentation task where we predict for each time step input whether there is an elephant call present, for which we developed a novel model based on a bidirectional LSTM, and (2) we also propose a novel trigger-call detection task where we predict the completion of an elephant call, for which we developed a lightweight LSTM model. We introduce significant improvements to our approaches by adapting different audio-specific techniques into our architectures and data-preprocessing. Our results demonstrate how our methods are of significant interest for conservation efforts. In particular, our bidirectional approach can reduce the need for human input for labeling and validating calls and the lightweight trigger-call approach has the potential to run a real time call-detector on low-cost remote devices.

1 Introduction

At an accelerating rate, human influence has taken a toll on the environment and the species it supports. Poaching, illegal logging, infrastructure development, and other human activities present great threats to wildlife and demand increased conservation efforts. In order to properly allocate resources and shape conservation, scientists must accurately model endangered species populations, migratory behaviors, and daily activities. For species that inhabit vast territories or exist in remote areas not easily accessible by humans, monitoring tasks become much more difficult. One promising solution is Passive Acoustic Monitoring (PAM), which involves placing autonomous recording devices throughout a habitat to passively record the vocalizations of species. Acoustic monitoring often serves as a cheaper and in many situations more practical data collection tool compared to, for example, visual traps because its range is not limited to direct line of sight. However, extracting meaningful semantics from large scale audio datasets present unique non-trivial challenges that, to be effective, require automation.

In this paper we focus our attention primarily on the African forest elephant. The African forest elephant represents a keystone species for the African rain forest, as it is responsible for the spreading and dispersal of many different seed species (Turkalo, Wrege, and Wittemyer 2017). Tragically, however, it is deeply threatened by human behaviors, such as poaching and deforestation, which have led to dramatic deceases in the population – over 60% in the last 100 years (Turkalo, Wrege, and Wittemyer 2017). Through the Cornell Lab of Ornithology, extensive acoustic data has been collected with the goal of better understanding population distributions, behaviors, and threats to survival (Bjorck et al. 2019).

Figure 1: African forest elephants gathering in a clearing.

We propose LSTM based models with architectural and data-preprocessing adaptations that improve the state of the art for detecting elephant calls on two separate tasks of interest to conservationists: **(1)** Per time-step prediction in an audio file for whether an elephant call is present (similar to a classical segmentation task) for which we developed a novel model based on a bidirectional LSTM. This task is extremely useful to conservationists who have hundreds of thousands of hours of audio data and need to segment out the sections with elephant calls locally. Without an automated approach conservationists have to resort to expensive and time-intensive human labelling. **(2)** A novel task concerning the detection of the conclusion of an elephant call, similar to the "trigger-word" detection task seen traditionally in systems such as the Amazon Echo that is awoken by the word "Alexa", for which we developed a lightweight LSTM. We propose this task because of its potentially additional importance to conservationist: with our lightweight model,

conservationists now have the ability to deploy low-power and low-cost listening devices that can selectively choose to store or transmit audio upon hearing the completion of an elephant call, allowing for more remote deployments at a lower cost. **(3)** We also introduce significant improvements to both models by adapting several different audio-specific techniques into the data-preprocessing and architecture that may be useful on other audio tasks. **(4)** *Our empirical results demonstrate how our approaches improve the state-of the art for elephant call detection.* Additionally, they show the effectiveness of our approach for addressing some of the challenges in the monitoring of wildlife conservation under limited resources. We show that the LSTM architecture is well suited for on-device fast inference because it does not need to keep a window of the previous so-many seconds to rerun a model over at each timestep yet still performs well.

2 Related Work

The field of bio-acoustics has placed great emphasis on the automatic detection of different species based on auditory vocalizations. Because sound waves attenuate less in water (i.e. sound can travel much farther) a great deal of work has been put into PAM of different marine species as well as birds, due to their distinct "songs"(Kahl et al. 2017; Grill and Schlüter 2017; Zhang et al. 2019). Previous work has ranged from "hand crafted" feature learning (Sahidullah and Saha 2012), to representation of rich distributions through Gaussian mixture models (Juang, Levinson, and Sondhi 1986), to more recent applications of deep models (Kahl et al. 2017; Grill and Schlüter 2017; Messner, Zöhrer, and Pernkopf 2018; Cakır and Virtanen 2017). A very common feature in these past works is the transformation of 1 dimensional raw audio into the more expressive 2 dimensional spectrogram equivalent (Bjorck et al. 2019; Zhang et al. 2019; Cakır and Virtanen 2017). Moreover, deep learning approaches have shown promise using this representation. For example, work by Zhang et al. used convolutions architectures to study the calls of blue and fin whales (Zhang et al. 2019) and Grill et al. utilize convolutional recurrent nets for classification of bird songs (Grill and Schlüter 2017).

Although significant research has been done into marine mammals and bird calls, there has been little research into mammals such as elephants. Elephant calls present a unique challenge because much of the richness of a call occurs in frequency bands below the level of human hearing (Elephant Listening Project, Cornell Lab of Ornathology 2019). In addition, calls are extremely infrequent in the audio collected, and often calls are layered on top of one another as multiple individuals will be in the same area at the same time.

Motivated by the initial success in elephant call segmentation by (Bjorck et al. 2019), we have made significant improvements on their results on the elephant call segmentation task by using more sophisticated base architectures as well as incorporating a variety of audio techniques into our models. We compared our approach against the previous state of the art (Bjorck et al. 2019), as discussed in the experimental section. Additionally, we propose a novel lightweight trigger-word elephant call detection task to address the conservationists' desire to deploy many low-cost low-power devices remotely into the forest and have the elephant call detection run locally.

Taking a broader perspective, we see that a great deal of research has been done into the general area of sound event detection. Different research has ranged from speech detection, to animal sound detection, to the detection of various noises that exist within the home (Cakır and Virtanen 2017; Lim, Park, and Han 2017; Wyse 2017; Kao et al. 2018). The DCASE (Detection and Classification of Acoustic Scenes and Events) challenge is a research driven challenge that looks to push the boundaries of acoustic detection and classification through different challenge tasks. We draw particular inspiration from the success of different models in the "Detection of rare sound events" challenge, as it mirrors in many ways the goal of our particular problem for both segmentation and trigger-word detection. Analyzing the top scoring models we see a common trend of combining convolutional and recurrent structure into the modeling of the problem (Cakır and Virtanen 2017; Lim, Park, and Han 2017; Wyse 2017).

3 Problem statement

We define the problem of automatic processing for PAM systems in two distinct ways: 1) as a segmentation problem where our goal is to classify each discrete time-step as being part of an elephant call or not and 2) as a trigger-word detection problem where we focus specifically on detecting the occurrence of an elephant call, while being less precise about its exact start and end time. More specifically, in the trigger-word detection setting we look to predict or "wake" our system on the time step directly after a call has occurred.

We consider these separate problem formulations for several reason. On the one hand, there is a great need for tools that can process large, pre-collected datasets and then output accurate time level segmentations. For example, the Elephant Listening Project has nearly a million hours of stored audio without the time or budget to hand-label all the calls in them. On the other hand, since these datasets are so large, there is great motivation to build devices that can on the fly detect whether an elephant call is likely to have occurred and, thus, selectively store audio frames. Moreover, if these devices could be made in such a way that they didn't require much power or complexity, then conservationists would be able to distribute them into more remote regions and at a lower cost. By framing the problem of elephant call detection as a wake-word detection problem, we propose a relaxed definition of detecting an elephant call (i.e. detecting its conclusion) to meet this objective, through the realization of light weight models defined on unidirectional time series data. Whereas in the case of the pre-collected audio segmentation task we can leverage more complex models that take advantage of both forwards and backwards processing, for real time processing/detection we must find smart ways to take advantage of unidirectional data.

4 Dataset

4.1 Elephant Listening Project

Established in 2000, the Elephant Listening Project (ELP) uses acoustic methods to study the ecology and behavior of forest elephants in order to improve evidence-based decision making concerning their conservation (Elephant Listening Project, Cornell Lab of Ornathology 2019). The ELP has gathered over 700,000 hours of data from around 150 locations within the African rain-forest, providing a very diverse and complete dataset for developing machine learning models. We are very thankful and lucky to have access to a rich subset of data collected between 2007 and 2012 from three sites in Gabon and one in the Central African Republic. Thanks to the tireless effort of experts and trained volunteers, hundreds of hours of audio have been labeled , where individual calls are labeled with their temporal duration (i.e. start and end time). As typical with PAM of large range species in a dynamic/chaotic environment (such as the rain-forest), elephant calls appear infrequently and many other sources of noise, both from other animals and human activity (logging, planes, cars), make accurate labeling a challenging task.

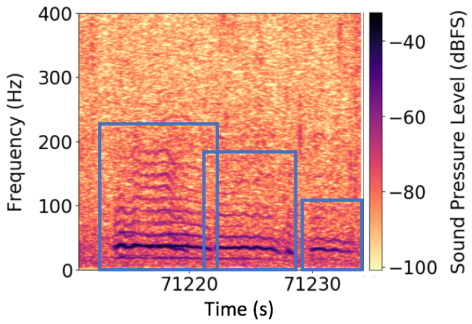

Figure 2: Example of three elephant calls.

4.2 Data Representation

Although the raw audio data is naturally one-dimensional, just as volunteers label/view the calls on a spectrogram, we leverage the rich descriptive features within the two-dimensional representation of sound. For a given raw audio sequence, we perform a short term Fourier transform to transform the data into a two-dimensional frequency by time representation. This data representation lends itself naturally to this problem because of the distinct acoustic characteristics of the elephant when expressed in this 2D "sound image:" namely stacked parallel "eyebrow" shaped lines, representing the harmonics of the fundamental frequency (see Figure 2). When computing the spectrogram representation, we use a window size of 512 and hop-length of 383 for transforming the audio signal into the frequency domain through the FFT. Additionally, we remove all frequency bands above 150Hz, as elephant calls rarely have significant signals above such frequency. Implemented on the down sampled audio data (1000Hz), each interval in the spectrogram encodes the frequencies within .512 seconds of raw audio.

Due to the large class imbalance within the data-set (depending on the location elephant calls make up only 0.7% to 20% of the data), we look to carefully facilitate a roughly homogenous data set to avoid majority class bias in training. Namely, for both of the proposed tasks, we extract fixed 25.5 time-step windows for each of the elephant calls, with the call randomly placed within. Because most calls range from length 5-10 seconds, we chose this window size to both capture significant elephant noise and background/adversarial noise. Based on this definition, we produce data samples that are 64 time steps by 77 frequency band tensors. For the segmentation problem, we define the ground truth labeling of each spectrogram time-step with a binary label indicating whether an elephant call occurs in that time-step. In contrast, for the trigger-word detection task we label 5 time-steps immediately after the end of the call with a positive 1 label. Although in trigger word detection we look to predict the end of the call, we soften this requirement (i.e. predict the end of the call within a specific interval) in order to help balance the training set.

When compiling our data set, because of the size of our data set, we employ a (95%5%) train/test split, resulting in $(106986, 5539)$ train, test examples for the segmentation dataset and $(105447, 5451)$ for the trigger-word detection set (different sizes are due to subtle differences in handling overlapping calls). We carefully paid attention to the drawing of the training and test data sets. In particular, we noticed that it is likely that the training and test data sets in (Bjorck et al. 2019) are biased towards having the elephant calls towards the middle of the selected set of training/testing audio segments. Furthermore, we also made sure that the test set would not be polluted by including windows that overlap with training examples (i.e. when two calls occur near each other) which we think also occurred in (Bjorck et al. 2019). Therefore, we created data sets by carefully randomizing the selection of the time windows for the training and testing data sets from different non-overlapping time periods.

4.3 Data Pre-Processing/Normalization

We consider several different data pre-processing techniques, which we later show have a very significant impact on model performance. Through research and experimentation we found that applying a logarithmic scaling to the spectrogram's amplitudes as in a decibel scale sharpened the spectrogram's visual contrast by compressing the range of values and highlighting the ratios between sounds.

In addition, we consider a collection of different data normalization techniques. Motivated by traditional image processing models, we tried normalizing each spectrogram image by the mean and standard deviation across the training set. Additionally, because of the frequency variant nature of audio, we define "feature-norm" as a normalization across each frequency band in the spectrogram. Lastly, we consider a domain specific norm that looks to normalize based on background noise while preserving important call information. Because the signals are very sparse, we consider normalizing by just the frames without elephant calls (i.e. back-

ground), again across each frequency. We found this final normalization to be the most effective as shown later.

5 Methods

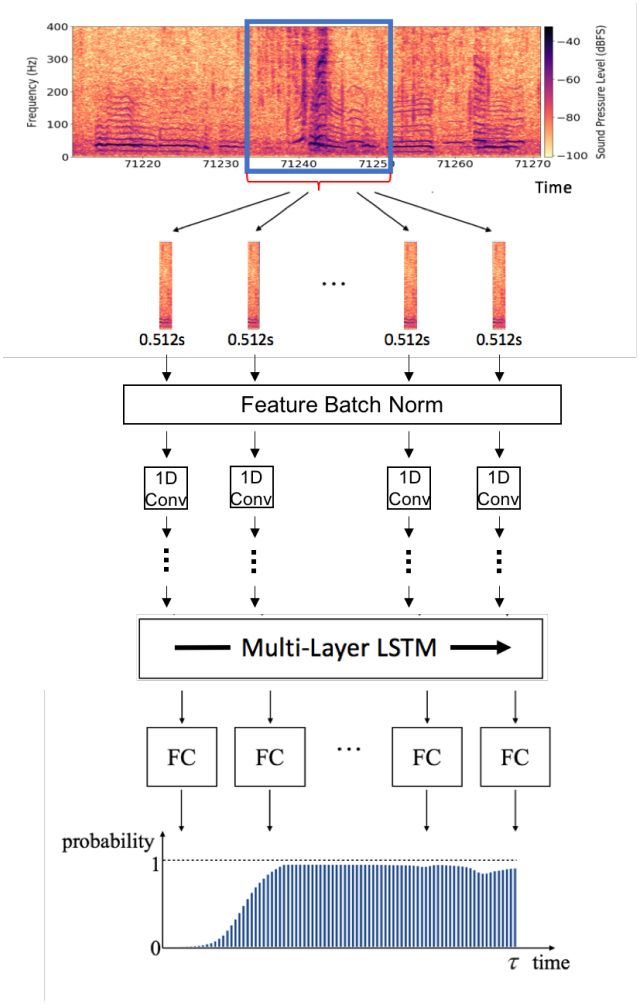

Figure 3: Overall framework for automatic call detection.

5.1 Unidirectional (Real Time) Models

Given that our task is to produce a label for each time step in real time and our models need to accept variable length inputs and keep track internally of previously seen information, the natural candidate model family are Recurrent Neural Networks (RNNs). RNNs process time step data sequentially, while keeping a representation of the past to help inform predictions at each state. This means that at inference time we don't have to process a sliding window of audio data for each new time step and can leverage the RNN's architecture and hidden state for efficiency. Traditional RNNs struggle to maintain long term information and so we use a Long Short-term Memory RNNs (LSTM), which has internal structure specifically built to promote longer term memory as well as to avoid gradient vanishing issues in training. We refer to this LSTM model as our baseline model.

5.2 Conv1D Layers with LSTM

Motivated by Lim et al. (Lim, Park, and Han 2017), we increased our model's representational power and accuracy by passing a time step slice first through one-dimensional convolution layers over the *frequency* domain, interspersed with maxpool, average pool, or a linear layer before inputting to the LSTM. While two-dimensional convolutions are commonly used in the literature for looking at temporal and spectral features, their usage is predicated on having access to the entire dataset chunk, which at least in our unidirectional problem statement is not possible. In addition, our research highlighted that 1D convolutions may be sufficient to achieve state of the art results. The 1D convolution works by sliding multiple 1D windows of weights and a bias over a single time step input. Each 1D window weight, or filter, leads to a single output channel and these are combined to produce an output of shape (num-filters by output of sliding weights operation). Since the Conv1D layer increases the input's size and even converts it from 1D (1 channel by num input frequencies) to 2D then we follow the convolutional layer with some form of a pooling or dimensionality reduction, where we experimented with a **1)** *flatten operation* which compresses the 2 dimensions into one long vector, a **2)** *maxpool layer*, an **3)** *average pool layer*, and a **4)** *linear layer*. These last 3 layers were run over the frequency dimension. Standard 1D maxpool reduces the size according to the filter size, and while we experimented with this we were motivated by (Lim, Park, and Han 2017) to use a maxpool with filter size equal to the convolutional output to effectively squash an entire dimension. An average pool employs a similar operation but rather than selecting the max element selects the average of the elements. And, lastly, applying a linear layer as a pooling operation we found was a very effective way to reduce dimension since its learnable parameters allow for a smarter averaging operation.

5.3 Linear Layers with LSTM

We found during our experiments that a linear layer worked surprisingly well as a dimensionality reduction operation. We also came across research that indicated a key difference in audio spectrograms from regular 2d images. While images in machine vision tasks are assumed to be translationally invariant, as in the distribution of values of a pixel do not depend on the pixel's location in the input image but rather its relative location to other important pixels, spectrograms are translationally invariant in the time direction but *not* the frequency dimension (Wyse 2017). This is because the meaning of an activation can drastically change depending on the frequency band it was produced in. For example, elephant calls are characterized by the low frequency rumbles discussed earlier. Our model is invariant to where a call appears in the time dimension since we process each time step feature separately and the LSTM is translationally invariant in its input features. However, by taking a 1d convolution and then maxpooling over the entire sequence as described, we realized that we were losing key frequency

information and our model accuracy suffered because of it. For this reason, we also experiment with more linear layers both as the previously described pooling operation and as an entire replacements of the convolutional operation.

5.4 LSTM Bidirectionality and Multiple Layers

For the segmentation problem, in which we had access to the entire data chunk at test time, we implemented the described above models but with bidirectional multilayer LSTMs. This bidirectionality is beneficial to the segmentation task because in a single direction predicting the presence of an elephant call takes "some time" while the multiple layers allow for more complex modelling. By running in both directions the model is able to process the call from both ends, allowing the model to make a more informed decision about the early time steps of the call as well as the end steps.

5.5 1D BatchNorm

We used a one-dimensional batchnorm over the frequency dimension on the inputs to allow our model to be more tolerant to differences in background noise across regions as well separate out interesting anomalies in a learnable loss driven pre-processing step. Effectively, this extra learnable scaling allows the model to learn to either completely undo our initial preprocessing or discover the most useful way to scale the training data. We use different parameters for each frequency in hopes of preserving the semantic differences between different frequencies as opposed to a normal image pre-processing operation in which we normalize by the mean and std over all pixels.

6 Evaluation

6.1 Segmentation Task and Trigger Word Task

We ran all of our unidirectional models on both the segmentation and trigger-word detection tasks. The bidirectional model variants were only applicable to the segmentation task since those at test time looked at time windows. We trained and tested on window chunks for both tasks for batching, and thus efficiency purposes, as well as to control the balance of labels our model saw so that we wouldn't be training mostly on background noise. *However, to more accurately evaluate our final models we ran the final models on the full uncut test data set.*

6.2 Optimization

For classification we used a standard Binary Cross-Entropy (BCE) loss and trained our models on Tesla P100 GPUs for anywhere between 5-50 epochs. For our optimization algorithm we used Adam since it typically requires less hyperparameter tuning and is resistant to noisy or sparse gradients. For most of our models the default initial learning rate of 1e-3 worked quite well with a l2 weight regularization of 1e-4. We experimented with learning rate decay and larger batch sizes than 32 but didn't find any clear performance improvements. We experimented with many different architecture hyperparameters and layer combinations and adjusted for bias/variance issues along the way.

6.3 Metrics

In line with standard procedure for classification tasks we used accuracy, precision, recall, and f-score. We considered using intersection over union as a possible metric but found that our precision and recall metrics captured that information enough and our following trigger word detection precision and recall were ultimately the most important.

6.4 Trigger Word Detection Precision and Recall

We also introduce metrics that help us optimize better for the ultimate use of our models in practice. We called these metrics trigger word detection precision and recall (*Trig P* and *Trig R*). We defined them respectively as the fraction of predicted elephant calls (not fraction of elephant call class labels) correctly labeled and the fraction of actual elephant calls that we predicted. By focusing on whether or not we indicated the existence of the call rather than how closely we captured the start and end times in the segmentation task or the number of arbitrary post wake word labels, we are able to evaluate our models better on the actual final task.

7 Results

We highlight model results that illuminate our understanding of model features as well as showcase the progression of performance. In Table 1, we give the results on the effect of the different normalization techniques discussed in section 4.3 for the baseline LSTM model, described in section 5.1, on the segmentation task.

Table 2 gives the performance comparison between the baseline model and our best performing models for the Trigger word and Segmentation tasks.

The four different model architectures are referred to by numbers $\{0, 7, 1, 9\}$, where we have (see also the Methods section 5): **Model 0** - Baseline LSTM from section 5.1; **Model 7** - BatchNorm 1D from section 5.5 followed by Conv1D Layer with *linear layer pooling* and 2-layer LSTM (see Figure 3 and section 5.2); **Model 1** - Conv1D layer with *flatten operation* and single layer LSTM (see section 5.2); **Model 9** - BatchNorm 1D followed by Conv1D layer with *flatten operation* and two layer Bidirectional LSTM (see section 5.4). All models have a final time distributed linear layer.

In Table 3, we provide the results when running our models models on the full uncut audio files. We see that in the Trigger-word detection problem this represents the process of real time detection; similarly for segmentation, this task models analysis of full uncut recording files.

8 Discussion

Overall, we obtained good performance on both tasks, and to our knowledge these results represent the state-of-the-art in this domain. After overcoming initial bias issues by properly normalizing our data set, using scaling, and adding feature-wise 1d batchnorm, we found that we reached a level of accuracy that was very hard to surpass. In trigger-word detection all of our unidirectional LSTM models regardless of number of linear, convolutional layers (with linear pooling), or filter sizes, after hyperparameter tuning, performed

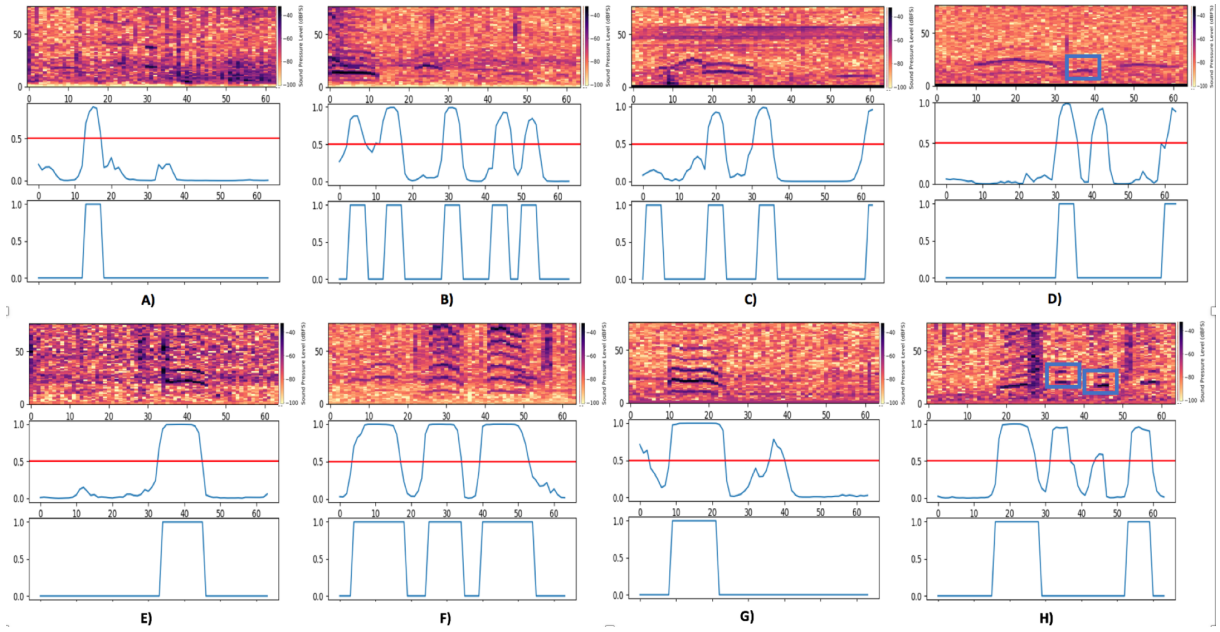

Figure 4: Visual representation of model performance. Top row - trigger-word detection; bottom - segmentation. **A)** Detect single tricky trigger. **B)** Detect many triggers, some obvious some not. **C)** Tricky missed trigger, one cannot see the call in the spectogram. **D)** Potentially unlabeled call due to labeler error identified in blue-box. **E)** Correct segmentation of a single call. **F)** Segmentation of several calls in a window. **G)** Incorrect segmentation of non-call (Note that it only makes a small mistake). **H)** Potential mislabeling identified by model.

Norm	Scale	Accuracy
None	None	62.02
None	Log	80.33
Standard	None	76.76
Standard	Log	86.57
Feature	None	82.42
Feature	Log	86.67
Background	None	82.67
Background	**Log**	**86.83**

Table 1: Different normalization techniques for the LSTM baseline model.

Model	Acc.	F1 (0)	F1 (1)	Trig R	Trig P
Trig-0	87.3	92.8	72.1	86.5	82.1
Trig-7	**89.3**	**93.0**	**72.7**	**85.73**	**86.4**
Seg-1	86.25	89.2	81.5	96.7	71.2
Seg-9	**90.16**	**92.0**	**86.9**	**94.2**	**86.1**

Table 2: Comparison of the baseline model (Trig-0 and Seg-1) with the best performing model (Trig-7 and Seg-9) for trigger word detection and segmentation. See the section 7 for a description of these models, and section 6.4 for definition of Trig R/P.

Model	Acc.	F1 (0)	F1 (1)	Trig R	Trig P
Trig-7	89.3	99.0	69.0	87.3	65.5
Seg-9	90.16	98.4	71.2	94.6	57

Table 3: Best model performances on uncut Test data.

similarly well. For both tasks, our models originally had a variance issue. We addressed this by early stopping and increasing L2 regularization. In the end, we seemed to still have a small bias problem; although our models were able to overfit (generally by 3% higher train accuracy than validation accuracy), the highest train accuracy was still not quite above 94%. From qualitatively analyzing our results we concluded that some of the model bias may be inherent in the dataset. For example, in examples D and H from Figure 4, there seem to be unlabeled elephant calls that our model correctly picked up on. In addition, on the correctly labeled elephant calls for example A, B, E, and F in Figure 4, while our predictions visually look very similar to the ground-truth, the accuracy and F1 metrics are potentially overly harsh to small insignificant deviations. *Overall, our models achieved high domain accuracy, precision, and recall for both problem definitions.*

We note the importance of the logarithmic scaling technique as well as normalization of the dataset. The particular norm we chose only has a minor overall effect, although models without an initial batchnorm layer were much more sensitive to this choice. This was somewhat surprising since

visually to us the different normalization techniques had a large impact on how easy the task would be. Overall, however, batchnorm seems to help our model tolerate more variable inputs which makes sense and it gave us a slight performance increase.

We also saw very interesting results using the different types of pooling after Conv1D layers. As we hypothesized, but in contradiction to the results of (Lim, Park, and Han 2017), max pooling with a filter size equal to the entire output in the frequency dimension performed poorly. Average pooling with the same filter size was slightly better but still poor. As mentioned previously, we hypothesize that this is because maxpooling over the frequency dimension treats the frequencies as translationally invariant, which may not be a good choice for a task like elephant call detection where the exact pitches of calls are very informative of their source. *Ultimately, the best results came from either flattening the convolutional layer outputs or using a learnable linear layer to weigh frequency features differently*. We decided to opt for linear pooling for its computational efficiency benefits.

As predicted the best performing model on the segmentation task was the bidirectional model that was able to take advantage of sequentially processing the data in both directions. We theorize that on the segmentation task it is difficult for a unidirectional LSTM to know immediately when to start labelling a time step since it requires a couple of moments for call confirmation, whereas the bi-directional LSTM is able to avoid this by processing the calls in both directions and combining results.

We should also mention that we did train our best models on the data sets from (Bjorck et al. 2019) and achieved a higher accuracy of 93.32% compared to their 91.71% on the segmentation task with our bidirectional LSTM. Nevertheless, as mentioned above, the results may be skewed, given their selection process of test time windows, as discussed earlier.

Lastly, looking at the performance of our best models on the uncut raw test audio (Table 3), we clearly observe our models' ability to generalize beyond the training setting. As expected, when applied to the raw test audio, the heavy class imbalance skews the performance of our model compared to Table 2. We see that our model has a tendency to overpredict elephant calls, leading to relatively low F1(1) and Trig P metrics; however, this is not necessarily bad as our models both do identify a large percentage of the true calls (Trig R). From a user perspective, this is what we want: Elephant calls are rare and we want our model to locate most of the calls.

9 Conclusion

We presented new state-of-the-art models for elephant call detection on the segmentation task and the newly defined trigger-call task. The high performance on the trigger word detection task using models with relatively few weights is particularly impressive. It suggests a potential future for real-time elephant call detection on low resource and inexpensive devices that could significantly improve scientists' ability to monitor these beautiful animals passively by deploying devices in remote areas. In the meantime, our segmentation task model will allow researchers to process the thousands of hours of audio data they have already recorded. We achieved these results through the adaption of several different audio-specific techniques into our data-preprocessing and model architectures that hopefully will be useful to researchers in other audio fields.

10 Future Work

A promising future direction entails better data-selection during training. For example, developing methods for more accurately capturing true class imbalances and more directed approaches to train on adversarial examples (such as chainsaw noises, car sounds, and other animal calls similar to that of an elephant) could have significant performance impacts.

Another direction is to apply our architectures to similar tasks across other animal call datasets. If successful, one can theorize it might be possible one day to have world wide automatic monitoring of animal and insect counts with unprecedented ease and scale. This would open up a number of biological and environmental research directions and help global conservation efforts.

References

Bjorck, J.; Rappazzo, B. H.; Chen, D.; Bernstein, R.; Wrege, P. H.; and Gomes, C. P. 2019. Automatic detection and compression for passive acoustic monitoring of the african forest elephant. *arXiv preprint arXiv:1902.09069*.

Cakır, E., and Virtanen, T. 2017. Convolutional recurrent neural networks for rare sound event detection. *Detection and Classification of Acoustic Scenes and Events (DCASE)*.

Elephant Listening Project, Cornell Lab of Ornathology. 2019. Elephant Listening Project. http://elephantlisteningproject.org/?__hstc=129768915.a2b4514dcaab4cfe838db856a9f8f4bd.1566351792825.1566351792825.1567373373497.2&__hssc=129768915.2.1567373373497&__hsfp=3448385070#_ga=2.242613376.329074020.1567373369-506375935.1566351792, Last accessed on 2019-08-29.

Grill, T., and Schlüter, J. 2017. Two convolutional neural networks for bird detection in audio signals. In *2017 25th European Signal Processing Conference (EUSIPCO)*, 1764–1768. IEEE.

Juang, B.-H.; Levinson, S.; and Sondhi, M. 1986. Maximum likelihood estimation for multivariate mixture observations of markov chains (corresp.). *IEEE Transactions on Information Theory* 32(2):307–309.

Kahl, S.; Wilhelm-Stein, T.; Hussein, H.; Klinck, H.; Kowerko, D.; Ritter, M.; and Eibl, M. 2017. Large-scale bird sound classification using convolutional neural networks. In *CLEF (Working Notes)*.

Kao, C.-C.; Wang, W.; Sun, M.; and Wang, C. 2018. R-crnn: Region-based convolutional recurrent neural network for audio event detection. *arXiv preprint arXiv:1808.06627*.

Lim, H.; Park, J.; and Han, Y. 2017. Rare sound event detection using 1d convolutional recurrent neural networks. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2017 Workshop (DCASE2017)*, 80–84.

Messner, E.; Zöhrer, M.; and Pernkopf, F. 2018. Heart sound segmentationan event detection approach using deep recurrent neural networks. *IEEE transactions on biomedical engineering* 65(9):1964–1974.

Sahidullah, M., and Saha, G. 2012. Design, analysis and experimental evaluation of block based transformation in mfcc computation for speaker recognition. *Speech Communication* 54(4):543–565.

Turkalo, A. K.; Wrege, P. H.; and Wittemyer, G. 2017. Slow intrinsic growth rate in forest elephants indicates recovery from poaching will require decades. *Journal of Applied Ecology* 54(1):153–159.

Wyse, L. 2017. Audio spectrogram representations for processing with convolutional neural networks. *arXiv preprint arXiv:1706.09559*.

Zhang, L.; Wang, D.; Bao, C.; Wang, Y.; and Xu, K. 2019. Large-scale whale-call classification by transfer learning on multi-scale waveforms and time-frequency features. *Applied Sciences* 9(5):1020.