# CS229 Final Project Report Identifying Elephant Vocalizations

## Introduction

The Asian Elephant has been worshipped for centuries and plays an important role in Asian culture and religion. It is also a key biological species in the tropical forests of Asia. In the wild, this majestic animal is threatened by extinction, mainly due to rapidly growing human populations. Wild elephant populations are mostly small, isolated and unable to join as ancient migratory routes are cut off by human settlements. These facts have led the World Wildlife Fund (WWF) to establish the elephant as a flagship priority species, and has determined that their survival cannot be guaranteed by conserving their habitat alone.

Being able to identify elephant vocalizations is a crucial part of understanding their communication patterns, which can then better inform decision makers about how to better conserve this iconic species. Thus, for this project, we aimed to identify and label a set of known Asian Elephant vocalizations. We used audio recordings of elephant sounds to build a series of classifiers with different models and features to attempt to capture the differences in these calls.

## **Dataset**

Our original dataset belongs to the Linguistics Data Consortium (LDC) at the University of Pennsylvania. Specifically, we used data set LDC2010S05 Asian Elephant Vocalizations [1], which consists of 57.5 hours of audio recordings of vocalizations made by Asian Elephants in the Uda Walawe National Park, Sri Lanka. These recordings feature several types of elephant sounds that have some social meaning collected over an 18 month period from primarily adult female and juvenile elephants. The raw recordings features long stretches of time during which no vocalizations can be heard, but about 31 hours of them have already been annotated with information on elephant calls, individuals, etc.

As for the elephant vocalizations themselves, researchers for this data set have identified 14 different calls. These fourteen classes are:

- Growl (GRW) - Bark-rumble (BRM) - Squeal (SQL)

- Squeak (SQK) - Trumpet (TMP) - Croak-rumble (CRM1) - Longroar-rumble (LRM) - Roar-rumble (RRM) - Chirp-rumble (CRM2)

Longroar (LRR)
Roar (ROR)
Musth chirp-rumble (MCR)
Bark (BRK)

## Features and Preprocessing

Before feature extraction, a fair amount of work went into preparing the data. The dataset came as a collection of FLAC files with the raw recordings, which had elephant calls interspersed. Additionally, the annotations for the recordings, when they had them, were in a separate text file in TextGrid format.

In order to prepare the data, we had to first use Praat [2], a linguistics software used in academia for these kinds of projects and that reads in sound files and TextGrid files. Using this program, we were capable of isolating several short recordings that each contained an elephant call.

After this, the files were culled for strange call labels (some annotations of the recordings had errors like unmentioned elephant calls) and extremely short recordings (where not enough data could be extracted from them). As an end result, we obtained a set of 4378 recordings with 13 different elephant calls (CRM1 and CRM2 were collapsed into a single class in annotation). The resulting data set was fairly skewed, with the following numbers of recordings for each type of call:

GWR	2864	SQK	420	LRM	236	RUM	192	LRR	148	TMP	141	BRM	139
RRM	75	ROR	46	SQL	43	BRK	36	CRM	29	MCR	9		

Having these recordings, we then proceeded to extract two separate feature sets in an effort to compare how each performed. For the extraction of the features themselves, we employed the openSMILE toolkit [3], a very flexible and powerful feature extractor for signal processing often used in speech recognition tasks.

#### Stats Feature Set

In order to classify sounds correctly, some temporal information must be included in the features. For this, we took after the approach of previous work in the field of emotion detection from speech, in particular the work of Jurafsky, et.al [4], where slices of audio where sampled and measurements like frequency, energy, etc. where used to predict certain attributes and emotions from speakers. Given that the vocalizations are not as complex as a human's, it seemed reasonable to simplify the task and use a more constrained set of features.

We focused on extracting the following from our elephant recordings: **Voicing probability** (a measure of, given a sound recording, how probable it is that it contains vocalizations); **Voice quality** (a feature used for human speech recognition that measures the quality of the fundamental frequency of the sound recording); **Frequency**, **Raw frequency** and **Smoothed frequency** (the fundamental frequency of the audio recording with thresholding, without and with some smoothing); **Energy** and **Log Energy** (a measurement of how much energy was present in the fundamental frequency and a good proxy for intensity of sound as well as its log value); **Jitter** and **Jitter delta** (a measurement of the differences between consecutive period lengths in sound waves [5] and its change over consecutive sound waves); **Shimmer** and **Shimmer delta** (similar to jitter, but measures the differences between consecutive sound wave amplitudes and its change along the waves).

Furthermore, for each of these features, we extracted a series of measurements, which were: average value, maximum, minimum, range, non-zero mean, standard deviation, skewness and kurtosis. Thus this dataset yielded an 80-dimensional vector of features per elephant recording.

#### MFCC Feature Set

The second feature set we used was modelled after another more commonly used approach to feature extraction for speech recognition. This consists of using mel frequency cepstral coefficients (MFCC), which are a time-frequency domain measurement of signals [6].

Put simply, these features are values that represent the spectrum of the log of the spectrum of the sound signal and are very useful in detecting phones. One important and useful property of MFCC coefficients is that, as a product of the process to extract them, each coefficient tends to end up being uncorrelated to the others. For speech recognition, the 12 first MFCC coefficients are generally used, and we emulate that approach here. In addition to these 12 values, MFCC features are often paired with a log energy measurement (for a total of 13 values) plus delta and double delta values of each (how they change over time). The results is a 39 dimensional vector that captures a lot of useful information about the sound signal for the time slice to which it corresponds.

For our approach, we wound extracting these MFCC vectors for every 10ms of sound in the recordings, with a smoothing of 25ms. This resulted in a large series of time-slice measurements for each recording. Because not all the recording were the same length, we decided to compress some of this information down in an effort to "normalize" the data. Thus, regardless of the length of the recording, we separated its MFCC vectors into 9 chronological sections and averaged them. This led to us having 351 features representing the 9 MFCC averages for the recording, at the expense of losing some of the details of the vectors being averaged.

#### **Models and Algorithms**

In our efforts to obtain the best possible results in our classification task, we used three different models on the feature sets we have just described. These models were:

## One-vs-many Logistic Regression,

Since logistic regression is a binary classification model, we prepared the data by first determining which type of sound we wanted the algorithm to identify and editing our labels vector by making the value be 1 when the label equaled the sound type we were focusing on and zero for all others. Then we ran logistic regression on our database for each type of sound.

#### Softmax Regression

After identifying each sound type individually, we implemented a more general multinomial classification algorithm. We then ran softmax regression with weight decay on the 13 classes of our database.

## Neural Network

Another model we implemented was a neural network with deep learning. The hope was that, given the power of these models, we could achieve better results than a more standard approach to classification could, in particular in light of the difficulty with dealing with the temporality involved in sound waves.

To this effect, we designed a simple neural network with one hidden layer. For the hidden layer activations, we used as our non-linearity the hyperbolic tangent. The output layer used the softmax function to normalize the results of our predictions for each of the 13 possible calls. Thus, the neural network prediction can be summarized in the following equation:

$$p_{\theta}(x^{(i)}) = softmax \left( U \cdot tanh(W \cdot x^{(i)} + b^{(1)}) + b^{(2)} \right)$$

## **Results and Analysis**

For our experiments detailed here, we used a training set with 3064 samples and a test set with the remaining 1314 samples. Initially, we used a random separation of the dataset into and later, to insure a more even separation, we took 30% of each class's recordings as part of the train set in order to ensure all classes were represented in both sets.

#### One-vs-many Logistic Regression,

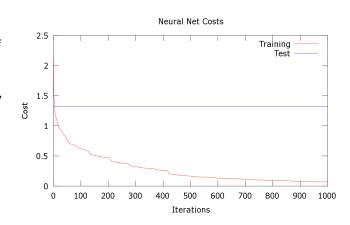
The effect of the significant skew in our data became very clear while using the logistic regression algorithm. It achieved its best result for the growl class, but all others would be nowhere as efficient. A more complete and balanced data set would help improve the success of this algorithm.

#### Softmax Regression

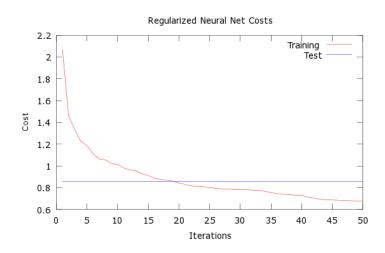
For the softmax regression model, the weight decay and learning rate were determinant in achieving better results. When using a smaller learning rate, the algorithm would converge quickly at a local minimum with high training and testing errors; while using a larger learning rate led to more accurate predictions.

## Neural Net

For our neural net, we initially ran it using the MFCC feature set. With our original implementation of the model, results were poor and very inconsistent, with the best run of the algorithm giving a 40% test accuracy. After improving the implementation, however, we ran the network once more. Using no regularization and a hidden layer of size H=50, we let the net run for 1000 iterations and got very promising results. With a training set accuracy of 98.86% a test set accuracy of 70.83%. The learning curve shown on the graph, however, very clearly illustrates a case of overfitting the training set.



Therefore, we set out to tune the neural net by using K-fold cross-validation (with K=10) to determine the best combination of hidden layer size and regularization parameter to minimize the test cost over 50 fixed iterations. The results show the best values are: H=200 and regularization parameter reg=0.3.



Using these parameters, we ran the neural net again to obtain the following results: training set accuracy of 81.02% and test set accuracy of 76.19%. As the graph on the costs over iterations shows, changing the parameters based on the model selection process we did and early stopping at the 50th iteration avoid overfitting the training set, and provide an improvement of 5.5% percent on the test set.

Digging deeper into these results, we can see the following values for accuracy, precision, recall and F1 for each of the classes in the following table:

	Examples	Accuracy	Precision	Recall	F1
GRW	859	0.951106	0.829442	0.951106	0.886117
SQK	126	0.706349	0.747899	0.706349	0.726531
LRM	70	0.514286	0.423529	0.514286	0.464516
LRR	44	0.272727	0.4	0.272727	0.324324
RUM	57	0.438596	0.5	0.438596	0.46729
BRM	41	0.04878	0.25	0.04878	0.081633
TMP	42	0.285714	0.48	0.285714	0.358209
RRM	22	0.045455	0.333333	0.045455	0.08
ROR	13	0	-	0	-
BRK	10	0	-	0	-
SQL	12	0.083333	1	0.083333	0.153846
CRM	8	0	-	0	-
MCR	2	0	-	0	-

In spite of the overall positive performance of the neural net, there are 4 classes we don't ever predict and our performance on each class diminishes with the number of examples we have for each one. Given that the training set has a similar composition of classes, these results, however, should not come as a surprise. The influence of having such a skewed dataset is going to be, naturally, that our network will learn to predict the more populous classes accurately and for the less represented ones, it is simply memorizing them from the training set, but not really learning how to predict them well. A larger data set with a more even amount of features, or reducing these data set by balancing the classes out should help address these issues.

As a complementary experiment, we ran the neural net on the Stats feature set. The results were very similar to what we obtained with our previous models, if only slightly improved, so we do not delve into the details, but include them in the summary table below.

Results Summary

Model	Dataset	Training Error	Test Error	N iterations	
Logistic regression (one-vs-many)	Stats	15%	43%	1000	
Softmax	Stats	15%	43%	1000	
Neural network (H=50, reg=0)	MFCC	1.14%	29.17%	1000 fixed	
Neural network (H=200, reg=0.3)	MFCC	18.98%	23.81%	50 fixed	
Neural network (H=200, reg=0.3)	Stats	27.59%	24.89%	200 fixes	

## **Future Work**

Further development of the project may help get even better results, including:

- Working on improving our feature sets to obtain better results and capture the temporality of data better, and run our models on all feature sets to have a better view of the advantages and disadvantages of each model and feature set.
  - Focusing on tuning our hyperparameters of the models.
- Seeking either more data or to balance the classes present in our current data set to obtain better predictors for all classes
- Attempting to adapt a known and powerful EM algorithm used in speech recognition to very accurately determine the words in a recording to the simplified task of identifying the elephant calls as if they were words being spoken by these animals.

## Conclusions

- Results show potential for solving this task, but also the need for more work on features and models.
- Despite the added temporality of the MFCC feature set, the fact that we are averaging over several time-slices causes some loss of information which is, in part, responsible for some of the errors present.
- Surprisingly, the different feature sets seem to give relatively similar results, which comes to show that it is not always so that the more complex and convoluted the models and features the better the results.
- Most of the errors observed stem from the issue of skewed classes (with over half of the data being of one type of call). In addition, other challenges are the quality of both the recordings (with some very low quality and noisy ones) and of the annotations (with some of them being imprecise).
- The features could be enriched by capturing some of the temporality involved in audio (i.e., the beginning, middle and end of a recording may be different) which may be the key to better classification
- Neural nets perform splendidly in memorizing the training set, but require some fine tuning to ensure better results at test time and avoiding overfitting our training data.

## References

- [1] LDC Asian Elephant Vocalizations dataset information and details can be found at https://catalog.ldc.upenn.edu/LDC2010S05
- [2] Praat software and tutorials available at http://www.fon.hum.uva.nl/praat/
- [3] openSMILE, software and manual, available at http://opensmile.sourceforge.net/
- [4] McFarland, D.A., Jurafsky, D. and Rawlings, C., "Making the Connection: Social Bonding in Courtship Situations", American Journal of Sociology 118, no. 6 (2013), pp. 1596-1649, 2013.
- [5] Farrus, M., Hernando, J. and Ejarque, P., "Jitter and Shimmer Measurements for Speaker Recognition", 2007
- [6] Jurafsky, D. and Martin, J.H., "Speech and Language Processing", Pearson Education, 2014 International Edition
- [7] Asian Elephants. WWF. Available at: http://wwf.panda.org/what we do/endangered species/elephants/asian elephants/