Statistical Learning-Classification

Project Title: Audio Classification of Cats and Dogs

Project Number: 9

Group Members:

Surname, First Name	Student ID	STAT 441	STAT 841	CM 763	Your Dept. e.g. STAT, ECE, CS
Yuan, Chen	20561973	■	□	□	STAT & CM
Amores, Angelica	20509683	■	□	□	STAT
Mehvee, Ammar	20542159	■	□	□	ACTSC & STAT
Yang, Cheryl	20505399	■	□	□	STAT

Your project falls into one of the following categories. Check the boxes which describe your project the best.

1. □ **Kaggle project.** Our project is a Kaggle competition.
   - This competition is active □ inactive □.
   - Our rank in the competition is ... .
   - The best Kaggle score in this competition is ... ..., and our score is ....
2. □ **New algorithm.** We developed a new algorithm and demonstrated (theoretically and/or empirically) why our technique is better (or worse) than other algorithms.
3. ■ **Application.** We applied known algorithm(s) to some domain.
   - ■ We applied the algorithm(s) to our own research problem.
   - □ We tried to reproduce results of someone else's paper.
   - ■ We used an existing implementation of the algorithm(s).
   - □ We implemented the algorithm(s) ourself.

Our most significant contributions are (List at most three):

(a) Perform a feature-wised analysis on audio sounds and compare between different classification algorithms.

(b) Identifying the best model that can be used to classify cat & dog audio sound with feature extraction.

(c) Identifying significant features related to animal sound classification.

List the name of programming languages, tools, packages, and software that you have used in this project:

- Languages: Python, R
- Packages: Librosa, Numpy, Pandas, Keras, os, sklearn, seewave, tuneR

Audio Classification of Cats and Dogs

Chen Yuan, Angelica Amores, Ammar Mehvee, Cheryl Yang

April 19, 2018

1 Introduction

In 2013, ImageNet launched a Kaggle challenge titled "Dogs vs. Cats" which focused on the problem of classifying images of cats and dogs. This popular challenge inspired the audio version of the Dogs vs. Cats classification problem, the Kaggle dataset "Audio Cats and Dogs". The goal of this project is to build and compare classifiers for this audio dataset using techniques typically used to extract useful features of audio data. Algorithms such as Adaboost, Random Forest, Logistic Regression with L_1 Regularization, Neural Networks, and Convolutional Neural Networks were explored.

2 Background

The topic of audio classification includes a wide array of problems ranging from music genre classification to speech recognition. Feature extraction is an important component of audio classification and usually involves selecting the spectral and temporal features pertaining to individual frames of an audio (i.e. very short segments of signal).

A spectrogram is defined as an intensity plot of the Short-Time Fourier Transformation (STFT). The STFT is essentially a Discrete Time Fourier Transformation (DTFT) on a sliding window. The time series is decomposed into a sum of finite series of sine and cosine functions. Each sine and cosine function has a specified frequency and a relative amplitude. The frequency and amplitude are used to build the frequency spectrum over each successive section over the time wave. A mel-spectrogram maps the spectrogram to the mel scale:

$$\text{mel} = 2595 \times \log 10 \left(\frac{1 + \text{hertz}}{700} \right).$$

The Mel-Frequency Cepstral Coefficients (MFCCs) drastically reduce the spectrogram into far fewer frequency bins as well as larger steps in time.

Chroma-STFT, Spectral Contrast, and Tonnetz extract features that are typically used for musical audio classification and analysis. In Chroma-STFT, the entire spectrum is projected using STFT onto 12 bins to represent the 12 distinct semitones (or chroma) of the music octave over time. Spectral Contrast considers the spectral peaks, which correspond with harmonic components; and spectral valleys, where noises often appear. Tonnetz is a planar representation of pitch relations, first attributed to Euler, and later used extensively by 19th century music theorists such as Riemann and Oettingen and in recent years by Neo-Riemanninan Music Theorists. Close harmonic relations are modeled by small distances on the plane.

3 Experiment

3.1 Data

The dataset being analyzed comprises of 280 .wav files containing either dog barks or cat meows. 167 files are categorized as cat meows, while 113 files are categorized as dog barks. Overall, there are 1,323.90 seconds of cat audio and 598.44 seconds of dog audio with the mean file length being 7 seconds. This data set contains a wide range of cat and dog sounds, covering different lengths of the animal utterances, lengths of silences, positions of animal utterances, and background noises which our models must account for. While contemporary audio classification focuses on the use of deep neural networks, the cat and dog classification problem is a binary one and the dataset is relatively small for building a classifier so other models may be more appropriate.

3.2 Preprocessing

To extract the useful features from the audio data, we used the Librosa library in Python. The audio files are loaded and decoded as a time series, represented as a one-dimensional NumPy floating point array. All audio files are mixed to mono and re-sampled to 22050 Hz. The methods described in the Background section were used: mel-spectrogram with a window length of 2048 and hop length of 512, MFCC, chroma-STFT with a hop length of 512, spectral contrast with window length of 2048 and hop length of 512, and tonnetz.

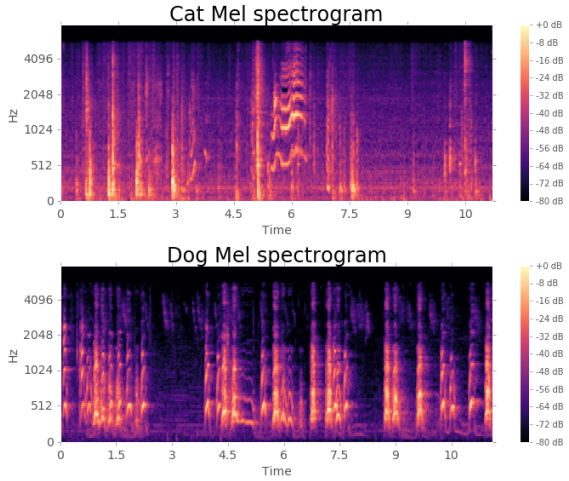

Figure 1: Mel spectrograms for one cat and one dog audio file.

Figure 1 is an example of mel-spectrograms of one cat and one dog file from the audio data set. Note that the cat file contains background sounds, making the spectrogram noisy. For the dog file, along with background noise, there are two individual dogs barking at the same time. Hence, an overlap of similar sound waves is observed in the spectrogram.

In total, 193 features are used by taking the average along the time axis and then stacked column-wise for each individual audio file to obtain a data set in the form of a NumPy array with dimensions 277×193, where each row corresponds to an audio file. The final dataset is then standardized and shuffled randomly.

3.3 Model Architectures

In order to compare the general performance of different algorithms, we used a nested cross-validation to prevent a potential leak of information from the test set to the model. The idea is to add a inner loop to the training set in order to optimize hyperparameters to that model, and then validate on the testing data. For the inner loop (training loop), we used a 3-fold cross validation, and for the outer loop (test loop), we used a 5-fold cross validation.

3.3.1 Adaboost

Models were tuned over the following hyperparameters:

- Learning rate: 0.1, 0.5, 1
- Maximum number of estimators at which boosting is terminated: 100, 200, 300
- Minimum number of samples required to split an internal node in decision trees: 6, 9, 2
- Max depth of decision trees: 1, 3, 5

3.3.2 Support Vector Machines

Models were tuned using the following hyperparameters:

- Kernel type: linear
- Penalty parameter C of the error term: 0.1, 1, 10, 25
- Tolerance for stopping criterion: 0.0001, 0.001, 0.01

3.3.3 Lasso

Regularization can be used in logistic regression to avoid overfitting, especially when there is only a small number of training samples, or when there is a large number of parameters to be learned. In particular, L_1 regularized logistic regression is often used for feature selection and have been shown to have good generalization performance in the presence of many irrelevant features [12] [4]. L_1 regularized regression is the optimization problem of

$$\min_{\theta} \sum_{i=1}^{M} -\log \left[p(y^{(i)}|x^{(i)};\theta) \right]$$

subject to $||\theta||_1 \leq C$ [10].

Models were tuned using the following hyperparameters:

- Normalization in the penalization: 'L_1'
- Inverse of regularization strength (is the same C used in equation above): 0.01, 0.1, 1,10
- Tolerance for stopping criteria: 0.0001, 0.001, 0.01

3.3.4 Random Forest

Models for each fold using GridSearch by tuning with the following hyperparameters:

- Number of trees in the forest: 10, 50, 100
- Minimum number of samples required to split an internal node: 2, 4, 8
- Number of features to consider when looking for the best split: 4, 8, 10
- Maximum depth of tree: None, 1, 3, 5

3.3.5 Neural Network

Models were tuned using Random Search with the following hyperparameters over 300 iterations:

- Hidden layer sizes: 50, 100, 200, 400, 600
- L_2 penalty: 0.001, 0.01, 0.1, 1.0

3.3.6 Convolutional Neural Network

The convolutional neural network models in the experiment had 4 layers with batch-normalization, max-pooling layers, and 1 fully connected layer with an output layer. Dropout was utilized during training after the second, fourth layers and fully-connected layers. Training was performed on a batch size of 28 and over 30 epochs.

3.4 Results

Measures used for the comparison of algorithm performances are overall accuracy, and measures regarding the ROC curve and AUC and are presented in Table 1 and Figure 2. The Adaboost, Random Forest, and Neural Network models performed the best in terms of overall accuracy. The convolutional neural network seemed to have the lowest performance out of all of them, but is not too far off.

Table 2 shows the results of Mcnemar's test for the models. It indicates that the two best models, Adaboost and Random Forest perform equally well. It also shows that SVM performs about as well as CNN.

	Adaboost	SVM	Lasso	Random Forest	Neural Network	CNN
Accuracy	0.90±0.04	0.85±0.03	0.88±0.02	0.90±0.03	0.90±0.03	0.86±0.04
AUC	0.96±0.03	0.91±0.04	0.94±0.03	0.95±0.03	0.94±0.03	0.88±0.06
Sensitivity	0.81	0.82	0.86	0.78	0.83	0.81
Specificity	0.96	0.87	0.90	0.98	0.95	0.88
PPV	0.93	0.82	0.86	0.97	0.92	0.83
NPV	0.88	0.88	0.90	0.87	0.89	0.87

Table 1: Performance of different algorithms.

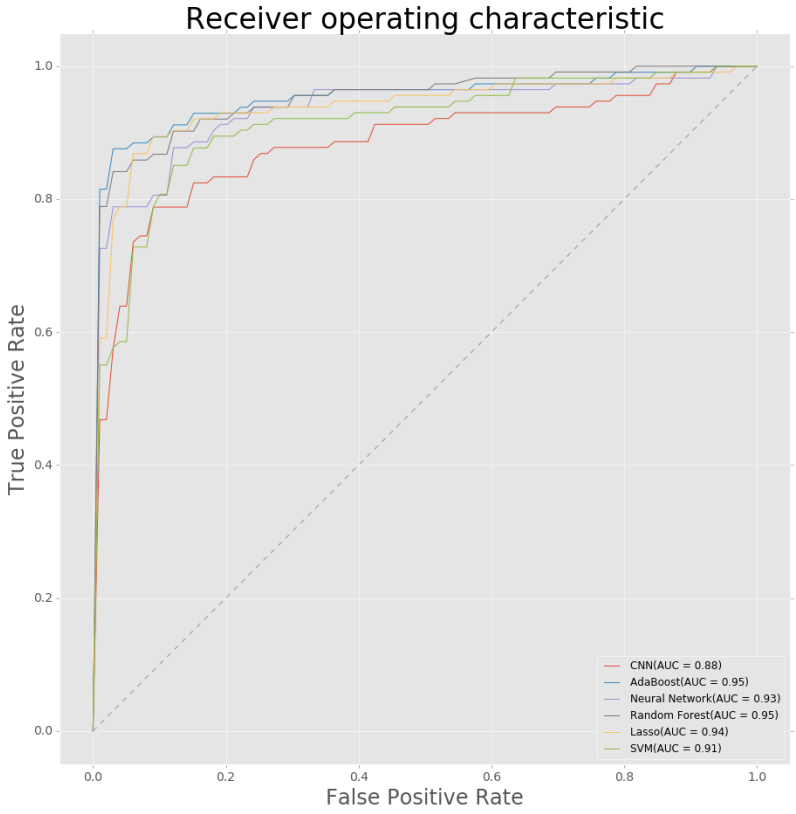

Figure 2: ROC and AUC comparison of different algorithms.

	CNN	Neural Network	Random Forest	Adaboost	SVM	Lasso
CNN	-	0.0192	0.1214	0.0614	1.0000	0.1516
Neural Network	-	-	0.6072	0.7744	0.0043	0.3833
Random Forest	-	-	-	1.0000	0.0708	0.8506
Adaboost	-	-	-	-	0.0428	0.7011
SVM	-	-	-	-	-	0.0352
Lasso	-	-	-	-	-	-

Table 2: Mcnemar's test comparing each pair of different algorithms.

4 Conclusion

Based on the test statistics we have, we choose Adaboost and Random Forest as our best algorithms for classifying cat and dog sounds.

The top 10 important features of Adaboost include MFCC, contrast, Mel-spectrogram and tonnetz. The top 10 important features of Random Forest include Mel-spectrogram and MFCC. In both of these models, Mel-spectrogram and MFCC are significant.

These algorithms can be applied to fields other than animal sounds classification, e.g., music plagiarism detection, electroencephalography eye blinks removal.

References

[1] Leo Breiman and Adele Cutler. Random forests leo breiman and adele cutler.

[2] Dan Ellis. Chroma feature analysis and synthesis, Apr 2007.

[3] Yoav Freund and Robert E. Schapire. A short introduction to boosting. *Journal of Japanese Society for Artificial Intelligence*, 14(5):771–780, Sep 1999.

[4] Joshua T. Goodman. Exponential priors for maximum entropy models, February 2007.

[5] Christopher Harte, Mark Sandler, and Martin Gasser. Detecting harmonic change in musical audio. *Proceedings of the 1st ACM workshop on Audio and music computing multimedia - AMCMM 06*, Oct 2006.

[6] Shawn Hershey, Sourish Chaudhuri, Daniel P. W. Ellis, Jort F. Gemmeke, Aren Jansen, R. Channing Moore, Manoj Plakal, Devin Platt, Rif A. Saurous, Bryan Seybold, and et al. Cnn architectures for large-scale audio classification. *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Jan 2017.

[7] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015.

[8] Dan-Ning Jiang, Lie Lu, Hong-Jiang Zhang, Jian-Hua Tao, and Lian-Hong Cai. Music type classification by spectral contrast feature. *Proceedings. IEEE International Conference on Multimedia and Expo*, Nov 2002.

[9] Serkan Kiranyaz, Toni Mäkinen, and Moncef Gabbouj. Dynamic and scalable audio classification by collective network of binary classifiers framework: An evolutionary approach. *Neural Networks*, 34:80–95, Oct 2012.

[10] Su-In Lee, Honglak Lee, Pieter Abbeel, and Andrew Y. Ng. Efficient l1 regularized logistic regression. *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI-06)*, page 401–408, Jan 2006.

[11] Marc Moreaux. Audio cats and dogs - classify raw sound events, Oct 2017.

[12] Andrew Y. Ng. Feature selection, l1 vs. l2 regularization, and rotational invariance. *Twenty-first international conference on Machine learning - ICML 04*, 2004.

[13] Karol J. Piczak. Environmental sound classification with convolutional neural networks. *2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*, Sep 2015.

[14] Aaqib Saeed. Urban sound classification, part 1, Sep 2016.

[15] Jerome Sueur. A very short introduction to sound analysis for for those who like elephant trumpet calls or other wildlife sound, Mar 2018.

[16] Jozef Vavrek and Josef Juhar. Multi-level audio classification architecture. *Advances in Electrical and Electronic Engineering*, 13(4), Nov 2015.