How Are You Feeling? Inferring Mood from Audio Samples

Joel Haynie
Department of Computer Sciences
University of Wisconsin, Madison
jhaynie@wisc.edu

Ankit Vij
Department of Computer Sciences
University of Wisconsin, Madison
vij2@wisc.edu

Amanpreet Singh Saini
Department of Computer Sciences
University of Wisconsin, Madison
saini5@wisc.edu

Eric Brandt
Department of Computer Sciences
University of Wisconsin, Madison
ebrandt@wisc.edu

Abstract

Classification of audio samples is an application of deep learning that is receiving considerable attention today. Our project involves investigating this specialized area of machine learning in detail. First, we explore the background and theoretical basis for audio sample classification using deep learning methods. Second, we implement and train a practical proof-of-concept application that classifies musical samples into one of four 'moods' and report test. Finally, we comment on emerging research topics in audio classification using deep neural networks.

1 Motivation and Theoretical Basis

Works we plan to cite (to make sure bibtex is working):

- CNN Architectures for Large-Scale Audio Classification, Hershey[7]
- What's wrong with CNNs and spectrograms for audio processing?, Rothmann[9]
- Inside the spectrogram: Convolutional Neural Networks in audio processing, Dorfler[3]
- Getting Started with Audio Data Analysis using Deep Learning, Shaikh[10]
- Hearing AI: Getting Started with Deep Learning for Audio on Azure, Zhu[14]
- How do deep convolutional neural networks learn from raw audio waveforms?, Gong[5]
- Learning from Between-class Examples for Deep Sound Recognition, Tokozume [11]
- AudioSet [6]
- TensorFlow [1]
- Keras [2]

1.1 Preprocessing

TODO

1.2 CNN Training

TODO

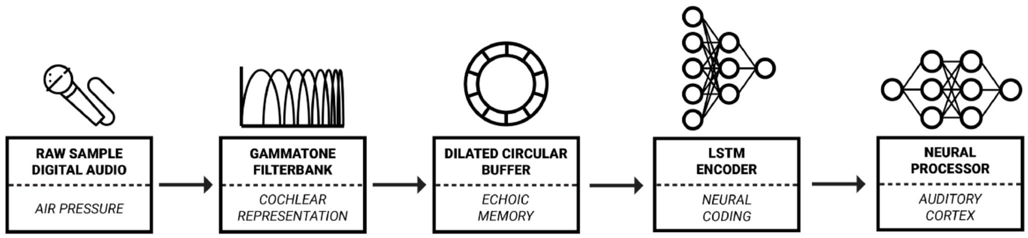

Figure 1: Rothmann's [9] ML model of human hearing

Flow.png

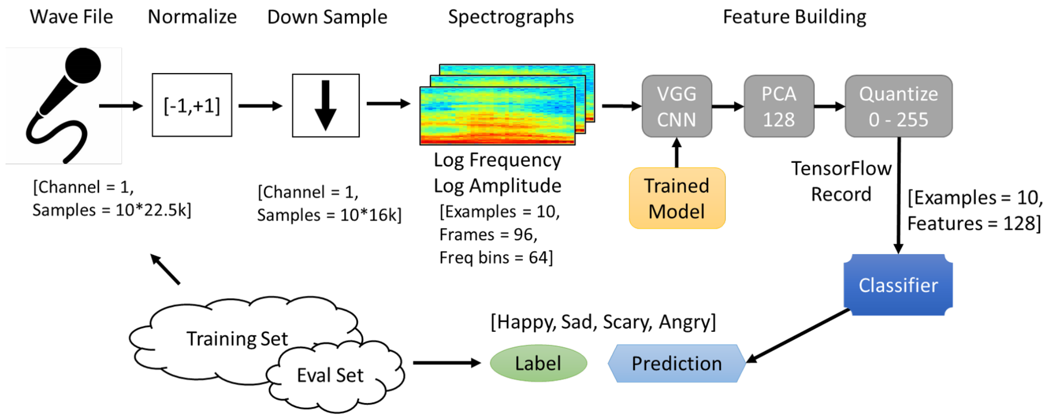

Figure 2: Hershey [7] model of converting `.wav` audio to features suitable for ML

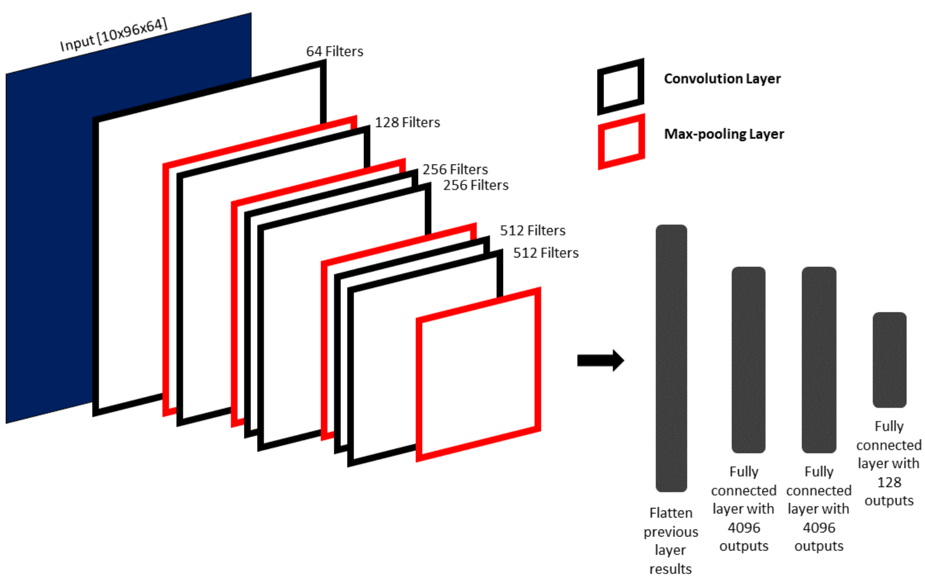

Figure 3: Deep network layers of the VGGish Convolutional Neural Network

2 Practical Implementation

2.1 Data Acquisition and Extraction

We collected our data from Google's AudioSet [6] which consists of an expanding ontology of 632 audio event classes and a collection of 2,084,320 human-labeled 10-second sound clips drawn from YouTube videos. The ontology is specified as a hierarchical graph of event categories, covering a wide range of human and animal sounds, musical instruments and genres, and common everyday environmental sounds. From the dataset, we focused on music mood samples and extracted data instances having labels of four mood classes- *Happy*, *Sad*, *Angry*, and *Scary*. The dataset selected was divided into two groups: training and evaluation.

To form our data sets, we filtered two disjoint sets from the ontology: First, a set of 400 instances with our corresponding labels for training. Of these 400, we ensured we chose 100 entries of each mood classification. The second set contained 223 data points with 56 instances per label which we used for evaluating our models. To avoid contamination, these 223 data instances were held aside during training and not introduced to the model until the evaluation stage. We created `.csv` files containing, for each instance, the YouTube video ID, start and stop times within the video, and the label associated with the instance. These `.csv`s were then used to run a batch processing script to download the audio samples in `.wav` format directly from their source (YouTube).

2.2 Transformation to Features

The next step in the application's pipeline is to transform the raw `.wav` files into features that can be input into a classifier network. This is done via the process described above in sections 1.1 and 1.2, using an open source TensorFlow model called VGGish [4]. We made initial attempts to implement this layer ourselves, but the training required for the CNN to achieve suitable results is beyond the computational capacity of our local workstations. In this instance, 'to see further, we stood on the shoulders of giants[8]' (in this case, Google) and used the pre-trained CNN model to feature-ize the `.wav` files. We converted each of our 400+223 `.wav` file instances using VGGish to 128x10 element feature vectors.

2.3 Classification

After initial processing of the wav files into feature vectors (matrices) of dimension 128x10 we investigated different models to perform the multi-class classification task of predicting the 'mood' of the music from the feature vector.

To evaluate different classification models, Python was used enlisting the libraries TensorFlow[1] and Keras[2].

The 400 training samples were evenly divided by class for stratified cross validation and used to train 3 different neural networks:

1. Simple multi-class logistic regression classifier
2. 1-Layer LSTM (Long-short term memory) recurrent neural network
3. 3-Layer LSTM (Long-short term memory) recurrent neural network

In each case, the model was trained using batches of 40 samples, randomized at each presentation, for a sufficient number of epochs to infer steady-state accuracy.

We evaluated the performance of each of the 3 models by two methods:

1. Validation set accuracy over an increasing number of epochs, to watch for over-training and the any generalization gap.
2. Evaluation on a Test Set of 220 never-seen-before data instances.

The performance of the training sessions is shown in figure 4.

After training, the evaluation on the 220-instance balanced test set, we observed the accuracies shown in table 1.

Figure 4: Baseline training performance for 3 models.

Table 1: Baseline accuracy on held-aside test set of 220 instances for 3 models.

Model	Accuracy
Logistic Regression	0.803
1-Layer LSTM	0.830
3-Layer LSTM	0.731

Finally, to make sure that our four chosen classes do not have an abnormal correlation between any combinations of classes, we also computed confusion matrices for the models. The confusion matrix for Logistic Regression (arguably the best performing classifier) is shown in table 2.

3 Conclusion

3.1 Discussion

After evaluating three machine learning models- Logistic Regression, 1-Layer LSTM, and 3-Layer LSTM on the evaluation set, we can draw some important conclusions. First, the input features sets (output of VGGish) must already be nearly linearly separable, as evidenced by the strong performance of the simple multiclass logistic regression classifier. Google's preprocessing of the raw audio waveforms into 10-frame spectrographs, including processing by Google's own CNN and PCA reduction clearly has produced data that is well separated without significant further processing. Evidence of the linearly separable feature data is supported by the fact that much more complicated non-linear classifiers (1-Layer LSTM and 3-Layer LSTM) do not yield better performance. It is also clear that complicated models with more parameters, particularly the 3-Layer LSTM take significantly more time to train. There is evidence that the LSTM models are subject to overtraining at higher numbers of epochs, as validation set accuracy decreases somewhat with increasing number of epochs. This leads us to believe that for the practitioner who wishes to write an audio-based ML application

Table 2: Confusion matrix for baseline logistic regression classifier of 223 test instances.

	Happy	Sad	Angry	Scary
Happy	45	9	2	1
Sad	11	39	2	4
Angry	0	1	53	4
Scary	0	7	3	42

and use AudioSet and VGGish as a 'black box', he or she is likely to do very well with only a simple linear classifier at the end. Finally, specific to our data set, the confusion matrix suggests, surprisingly, that 'Happy' and 'Sad' are the most often confused classifications, and that 'Scary' and 'Angry' are comparatively easy to predict.

3.2 Future Work

In the future, we plan to extend the capabilities and accuracy of our model and by including recent developments taking place in the field of sound recognition. Tokozume et al. [11] talk about Between-Class learning (BC learning) which is a novel learning method for deep sound recognition. In a similar direction to overcome undesirable behaviors such as memorization and sensitivity to adversarial examples by neural networks, Zhang et al. [13] propose a simple learning principle called mixup which trains a neural network on convex combinations of pairs of examples and their label. Creating a new feature space by combining sounds from different classes, like suggested by BC Learning and Mixup, would not only provide variations in feature space, remove sensitivity to adversarial examples, but will also regularize the positional relationship between the class feature distributions. The results from the two papers also show that the model provides better performance, so exploring this direction could provide better accuracy in mood detection from audio.

Another direction that is also very appealing is the use of speaker adaptation and regularization techniques for neural networks for sound recognition. Tomashenko et al. [12] propose a novel way to improve speaker adaptive training for neural network acoustic models using GMM derived features for automatic speech recognition. Taking inspiration from this work, we can also extend our system to detect mood from user's speech using the techniques as suggested by Tomashenko et al. which will improve its capabilities of mood detection beyond music.

Additionally, the detection of mood from audio signals can be put to use for various recommendation systems. One possible application is the mood based song recommendation service and another could be targeted product recommendation based on inferred mood of the listener. There could be other possible applications like depression detection systems based on the user's choice of audio playlists etc.

References

[1] Google Brain. Tensorflow: An open source machine learning framework for everyone, 2018. URL `https://www.tensorflow.org`.

[2] François Chollet. Keras: The python deep learning library, 2015. URL `https://keras.io`.

[3] M. Dörfler, R. Bammer, and T. Grill. Inside the spectrogram: Convolutional neural networks in audio processing. In *2017 International Conference on Sampling Theory and Applications (SampTA)*, pages 152–155, July 2017. doi: 10.1109/SAMPTA.2017.8024472.

[4] Dan Ellis, Shawn Hershey, Aren Jansen, and Manoj Plakal. Vggish tensorflow model, 2018. URL `https://github.com/tensorflow/models/tree/master/research/audioset`.

[5] Yuan Gong and Christian Poellabauer. How do deep convolutional neural networks learn from raw audio waveforms?, 2018. URL `https://openreview.net/forum?id=S1Ow_e-Rb`.

[6] Google. Audioset: A large-scale dataset of manually annotated audio events, 2018. URL `https://research.google.com/audioset/index.html`.

[7] Shawn Hershey, Sourish Chaudhuri, Daniel P. W. Ellis, Jort F. Gemmeke, Aren Jansen, Channing Moore, Manoj Plakal, Devin Platt, Rif A. Saurous, Bryan Seybold, Malcolm Slaney, Ron Weiss, and Kevin Wilson. Cnn architectures for large-scale audio classification. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2017. URL `https://arxiv.org/abs/1609.09430`.

[8] Isaac Newton. Quotes attributed to isaac newton, 1727. URL `https://en.wikiquote.org/wiki/Isaac_Newton`.

[9] Daniel Rothmann. What's wrong with cnns and spectrograms for audio processing?, 2018. URL `https://towardsdatascience.com/whats-wrong-with-spectrograms-and-cnns-for-audio-processing-311377d7ccd`.

[10] Faizan Shaikh. Getting started with audio data analysis using deep learning (with case study), 2017. URL `https://www.analyticsvidhya.com/blog/2017/08/audio-voice-processing-deep-learning/`.

[11] Yuji Tokozume, Yoshitaka Ushiku, and Tatsuya Harada. Learning from between-class examples for deep sound recognition. *CoRR*, abs/1711.10282, 2017. URL `http://arxiv.org/abs/1711.10282`.

[12] Natalia A. Tomashenko, Yuri Y. Khokhlov, and Yannick Estève. Speaker adaptive training and mixup regularization for neural network acoustic models in automatic speech recognition. In *Interspeech*, 2018.

[13] Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *CoRR*, abs/1710.09412, 2017. URL `http://arxiv.org/abs/1710.09412`.

[14] Xiaoyong Zhu, Max Kaznady, and Gilbert Hendry. Hearing ai: Getting started with deep learning for audio on azure, 2018. URL `https://blogs.technet.microsoft.com/machinelearning/2018/01/30/hearing-ai-getting-started-with-deep-learning-for-audio-on-azure/`.