AudioVision: Sound Detection for the Deaf and Hard-of-hearing

Kyle Hipke
University of Washington
(redacted)
kwhipke1@gmail.com

Meleigha Holt
University of Washington
(redacted)
meleigha@gmail.com

Dylan Price
University of Washington
(redacted)
the.dylan.price@gmail.com

Amy Brumet
University of Washington
(redacted)
amybrumet@gmail.com

Kevin Anderson
University of Washington
(redacted)
kevinand@cs.washington.edu

ABSTRACT
We present AudioVision, an Android application that allows deaf and hard-of-hearing users to detect important, user-defined sounds through an accessible audio visualization. Using interactive machine learning, AudioVision users can create a customized sound recognition system and dynamically adapt it to their environment and needs. AudioVision provides visual feedback of detected sounds so that users can quickly glance at a notification to see what happened, and also so that they can recognize sounds that have been misclassified or are not part of the learning problem. In a user interview, we observed that the deaf user was able to use this visualization to easily comprehend qualities of the visualized sounds that enabled them to recognize their class. The entire AudioVision system is not finished, but is fleshed out enough that its completion is just a matter of further implementation.

Categories and Subject Descriptors
H5.2 [**Information interfaces and presentation**]: User Interfaces.

General Terms
Design, Human Factors.

Keywords
Interactive machine learning, accessibility, deaf.

1. INTRODUCTION
Many important events in the home produce sounds and require hearing in order to be detected. These sounds include doorbells, knocks, microwave and oven timers, fire alarms, dryers, washing machines, dishwashers, objects falling, baby cries, and dog barks.

The deaf and hard of hearing currently have no general purpose mechanisms for begin notified of such events. Special purpose systems for specific events exist, such as the lights in the household flickering when the doorbell rings, but there is no system which covers a wide range of sound events and is able to adapt to the unique cases of each situation.

In this paper, we describe Sound Detector, a system for learning, notifying, and visualizing sound events in the home. This system takes advantage of the ubiquity of affordable smartphone technology, current work in sound classification using machine learning techniques, and our own research in sound visualizations for the deaf and hard of hearing.

Sound Detector is currently just a proof-of-concept prototype and further work is required before it could be of practical use to the deaf and hard of hearing communities. We hope that others will be able to build upon our work and create useful tools for detecting sound events in the home.

2. RELATED WORK
2.1 Perception of sound by deaf and hard-of-hearing
We researched how the deaf perceive sound to better understand what role sound plays in a silent world and how sound is understood: [6] illuminated how Tucker came to navigate a hearing driven world. She learned at an early age to read lips and speak at an appropriate volume. But spent many years trying to interpret body language and context to replace information rich sounds. Tucker's story enforced how valuable the location and the source of a sound is to interacting with the world around us.

AudioVision borrows from the work of Lumisonic [8]. Lumisonic attempts to provide a real-time visualization of audio to enable deaf and hard-of-hearing users to perceive music. AudioVision makes use of this visualization idea to present a non-real-time display of audio to its users.

2.2 Interactive Machine Learning
AudioVision draws upon and contributes to the body of work on interactive machine learning. Like prior work in other domains (e.g., [1,2,3,4]), AudioVision allows end users to create custom classifiers by iteratively creating examples, using examples to train a classifier, evaluating the classifier, and adding or deleting training examples to influence classifier behavior. AudioVision provides a new end-to-end application of these techniques in an accessibility-oriented context. AudioVision also presents a unique focus on the challenge of supporting exploration and comparison of alternative learning concepts in a This is relevant to other real-world applications where the goal is to create a model tailored to the given context (the user's home), as opposed to a traditional focus on modeling a fixed "ground truth".

2.3 Sound Detection

AudioVision provides functionality similar to the Sound Detector app [9]. Sound Detector provides the ability to alert the user when audio at a phone goes above some user-defined threshold. However, it does not provide training and classification of the noises that it detects.

3. DESCRIPTION OF SOLUTION

3.1 Overall Flow of Application

3.1.1 Training

The first step is training a phone (the detector phone) to detect sounds in a room. The user records a set of examples for each sound they wish to detect, and gives these sounds a name. These examples are sent to the server, where a classifier for that phone is built. If the user decides to add more sound examples later these will also be sent to the server and the classifier rebuilt.

3.1.2 Detecting

The flow of sound detection starts with the detector phone. A sound event is detected and a short sample of audio is sent to the server. The server extracts features from the sample, loads the classifier for the detector phone, and runs the classifier on the features. The user phone is notified that a sound has been detected and it retrieves the data pertaining to the sound event from the server. It then notifies the user with a vibration and displays the name (location) of the detector phone, the classification for the sound, and a visualization of volume peaks in the sound.

3.2 Machine Learning Method

For classification of audio samples, we used a machine learning algorithm on a set of features extracted from the audio signals. For feature extraction, we chose the open source jAudio library ([7]) and for machine learning we used the open source Weka library ([10])

To process an audio sample, we first (using jAudio) resample the sound to 24000 Hz and normalize it relative to the other audio samples such that the maximum and minimum values for the signal are 1 and -1. Then, also using jAudio, we divide the sample into windows which are 32 samples in length, with an 8 sample overlap between windows. The following features are extracted from each of these windows: Magnitude Spectrum, Power Spectrum, Spectral Centroid, MFCC, Root Mean Square, Strongest Frequency Via Spectral Centroid, Zero Crossings, Spectral Flux, Strongest Beat, Beat Sum, Strength Of Strongest Beat. See jAudio's documentation for a description of each of these features and the corresponding algorithm Then we take the mean and standard deviation of these features across the entire sample and these values are used as the inputs to the machine learning algorithm.

For machine learning we used the Adaboost M1 algorithm implementation in the Weka library, using a decision stump as the base classifier.

This configuration for feature extraction and machine learning was found based on a test set of microwave, doorbell, and music/conversation sounds of approximately 5 seconds in length. A total of 94 samples were recorded on an Android Motorola Droid, and we used a randomly selected training set of 10 microwave, 10 doorbell, and 10 music/conversation sounds and the rest of the 64 sounds as a test set. With this test and training set we were able to achieve a 96.9% success rate, correctly classifying 62 out of 64 test sounds.

However, in the real system we were not able to achieve this same success rate. We believe that tweaking our peak detection algorithm, sample length, and increasing the number of required samples per sound would allow us achieve this, it is just a matter of more thorough and rigorous testing.

3.3 Visualization Design

Our project aim was to classify sounds so the user could be alerted when visual cues were unavailable. For example, a clattering stove fan that was left on after cooking or a knock at the door when the user was upstairs. We knew that identifying sounds with 100% accuracy was improbable so we needed to devise an alert that was representative of the qualities of the sound such that the user could guess what was occurring.

In parallel to researching how sound is perceived by the deaf, we looked into the various ways that sound has been visualized to both the hearing and deaf. The most common interpretation qualifies sound by frequency.

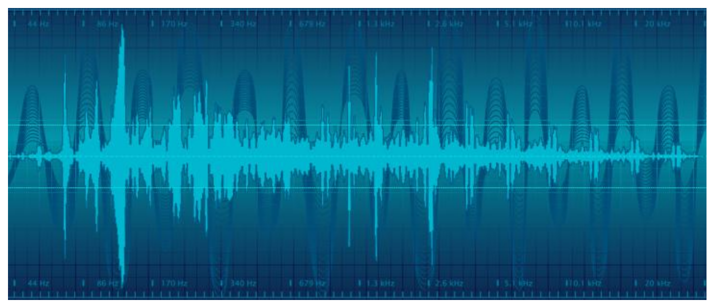

Image 1: Frequency Visualized

We wanted to challenge ourselves to show more of the qualities than just frequency that make a sound unique. The work of Evan Grant and Mick Grieson, researchers in the field of sound visualization, was inspiring. They work in a field called Cymatics, which deals with making sound visible through vibrations. Often Cymatics are demonstrated with a thin layer of powder on a metal surface. A speaker or other method of vibration is applied against the metal plate, transforming the powder into spectacular patterns.

A project called Lumisonic applies the cymatic's graphic potential of sound to an audience of deaf and hard of hearing ([8]). The goal of this application is to allow those who cannot hear manipulate sound. To make this possible, Lumisonic makes sound visual. The program is available for download but our team was unsuccessful in making the program work

Image 2: Lumisonic Visualization

Kristi Winter, assistant professor in American Sign Language at the University of Washington was our test subject for Audio Vision. Kristi met with us once at the beginning of the quarter and gave us insight into her experiences with deafness. From this preliminary interview, we aimed to identify important sounds. The notes from the interview are as follows (written from her perspective):

- It is important to be warned about danger. I rely on people's body language to understand where a sound/the danger is coming from.
- I would imagine it would be important for deaf mothers to know if their baby was crying. Right now, we have monitors with flashing lights but if you are looking away or are asleep, these lights are easy to miss.
- I want the application to tell me more than "there is a sound." Even if this means false positives, I am okay double checking to make sure everything is okay. Better to be safe.
- When I travel, there are more possibilities of un-expected and unfamiliar sounds. For example, in the mid-west I know there are tornado warnings which I am not used to dealing with in Washington. Sirens in general are important to know about but in the case of a tornado siren, there is no light to tell me about the sound.

At the end of the quarter, Kristi graciously agreed to meet with us again and test our application prototype. With the aid of interpreters and a script, we explained the tasks we wanted her to test. The script and notes taken are reprinted in the next section.

3.3.1 *Script*

This system requires more than one phone. There is a User Phone and one or more Detector Phones.

User Phone: the phone you carry with you everyday. This notifies you of a sound and displays a visualization with information about the sound.

Detector Phone: This phone listens for sounds from an assigned a location. Upon set-up of the system, you will teach this phone which sounds to listen for.

(Tip: The phone is more accurate in identifying a sound the more examples of the sound you record for it.)

At first, she understood this to mean that she carried with her two phones. One to detect sounds and another to alert her. We explained further that the detector phone was to be left in a permanent location in the house.

Step 1:

Set-up Detector Phone: You have just downloaded the app. Teach the phone which sounds are important for you to be notified about.

As she walks through set-up: "This is simple." "The way it's set up is great."

Some unfamiliarity with the Android interface. She is an iPhone user.

In this scenario, pretend the phone will be left in the **Living Room** and you want to know when someone **knocks** on the door.

To record knocks, she is confused if she presses the green bar to record or the red record button.

Step 2:

We are going to show you two visualizations of sounds. One represents and alarm and another represents someone knocking. Which sound do you think goes with which visualization?

First Visualization (door knock): "Is that a door knock? The pulse of it looked like how it feels to knock."

Second Visualization (alarm): "Since it's red, it must be an

alarm."

For both sounds she likes the accuracy indicated at the bottom.

3.4 Visualization Implementation

On Android, programming the visualizations required creating a custom View that takes the list of data points given from a Notification and stores them so it can animate when the visualization is drawn. Because Android doesn't support xml animations without preloaded drawables the animations were done programmatically by overriding the View's onDraw method and using the system clock to choose which frame of the animation to draw.

4. Future Work

In addition to finishing the development of this application, which requires improving general stability, there are several avenues future work could explore.

Accurate classification is extremely important for the usefulness of this app. Improvements could be made in this direction by experimenting with alternate classifier algorithms, audio features, and peak detection algorithms. The current peak detector doesn't distinguish individual peaks accurately enough to reliably identify periodic sounds (like knocks), so improvements to it would have a marked increase on AudioVision's utility.

It's difficult for users to actually produce some sounds that would be useful to identify, like gunshots, babies crying, and so on. A pre-loaded (perhaps crowdsourced) library of common sounds would, in addition to solving this problem, make it much easier for users to use the app (because they wouldn't need to give examples). However, care would need to be taken to ensure that the library of sounds works in the various acoustic environments the users will be placing detector phones in

Finally, in order for users to successfully build useful classifiers, they need some sort of feedback so they can rapidly iterate and modify the learning problem. If the application had some way of communicating the overall accuracy of the current classifier, and a quick way to edit the learning problem, users would be more likely to create more accurate classifiers.

5. REFERENCES

[1] Amershi, S., Fogarty, J., Kapoor, A., and Tan, D. Examining multiple potential models in end-user interactive concept learning. *CHI'10*, (2010), 1357-1360.

[2] Fiebrink, R., Cook, P.R., and Trueman, D. Human model evaluation in interactive supervised learning. *CHI'11*, (2011), 147-156.

[3] Fogarty, J., Tan, D., Kapoor, A., and Winder, S. CueFlik: Interactive concept learning in image search. *CHI'08*, (2008), 29-38.

[4] Hartmann, B., Abdulla, L., Mittal, M., and Klemmer, S.R. Authoring sensor-based interactions by demonstration with direct manipulation and pattern recognition. *CHI'07*, (2007), 145-154.

[5] Rosen, Russell S. "Representations of Sound in American Deaf Literature." *Journal of Deaf Studies and Deaf Education*. 12.4 (2007): 552-565. Print.

[6] Tucker, Bonnie P. *The Feel of Silence*. Philadelphia: Temple University Press, 1995. Print.

[7] "jAudio". *Java Audio Feature Extraction Library.* http://sourceforge.net/projects/jaudio/),

[8] "Lumisonic". *Audio Visualization Software.* http://soundandmusic.org/projects/lumisonic/downloads

[9] "Sound Detector". *Android application.* https://play.google.com/store/apps/details?id=dk.mvainformatics.android.sounddetector

[10] "Weka". *Machine Learning Library.* http://www.cs.waikato.ac.nz/ml/weka/