Joseph Standerfer (josephst)
Project 2 Writeup
Date: 10/8/2019

Project 2: Audio Classification

INTRODUCTION:

In this project, I explore several ML pipeline techniques for classifying audio samples. The 6 categories used for the study were song, microwave, blender, vacuum, fire alarm, and silence. The 4 different machine learning pipelines I created only differ in their methods for feature extraction. Each model uses a combination of spectrogram bins or domain features created from either a window (partial file) or full audio file. When windows were used for the classification process, I gathered the audio file prediction from a majority vote of the window predictions. All the pipelines achieved comparable levels of success, although the use of windowing gave slightly better results overall.

DATA COLLECTION:

The 6 categories of audio used in this project were all recorded on my phone. I used the app "WAV Recorder" to collect 20 (30 second) samples from each of the sources. Below I detail the exact collection method used for each source.

- Song
  - My 20 song sound samples were recorded from 4 different songs, comprised of 3 different genres of music. Each of the songs was recorded 5+ times by my phone while they were played on a JBL speaker. Also, each of the song recordings were taken at different intervals within the full song. My playlist is shown below:
    - Folsom Prison Blues - Johnny Cash
    - Cocoa Butter Kisses - Chance the Rapper
    - 3005 - Childish Gambino
    - Alive with the Glory of Love - Say Anything
- Microwave
  - The microwave recordings were taken from a microwave I have in my apartment. From each of the samples, I recorded myself placing an item into the microwave, closing the door, and selecting a time (20 – 35 seconds). After the cook was complete, I would continue recording until the alarm had sounded and I had removed the item from the microwave.
- Blender
  - The blender recordings were also taken from a single appliance. During the recordings, I blended a concoction of vegetables and peanut butter. I found these ingredients still held a good consistency even after several minutes of blending. To create some variation between samples, I added and subtracted ingredients as well as moved my phones position.
- Vacuum
  - The vacuum readings were all from a single vacuum. To create variation, I recorded myself using it on a few different carpets and while sweeping it around.
- Fire Alarm

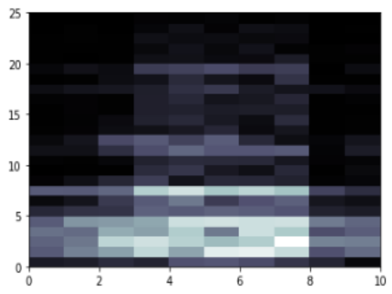
    - My recordings of a fire alarm all came from a single YouTube video (https://www.youtube.com/watch?v=nIfIAgkB-MI). This clip was played repeatedly on my laptop and recorded by my phone. I changed my volume and shifted the phone between each recording to cause some variation between the files.
- Silence
    - The silent recordings were all taken within my apartment. As you may note, they are not completely devoid of sound. I allowed there to be some noise from the outside streets. I felt this was a good proxy for what a person would consider silence as opposed to the complete lack of sound. The background noise varied between samples but usually consisted of muffled traffic, construction sounds, and air conditioning cycles.

FEATURE ENGINEERING:

In this project I used 2 different methods of feature extraction, spectrogram binning and domain features. Each of these methods was applied to either the full audio clip or a series of 3 second windows (with 50% overlap), to create 4 separate ML pipelines.

Spectrograms:

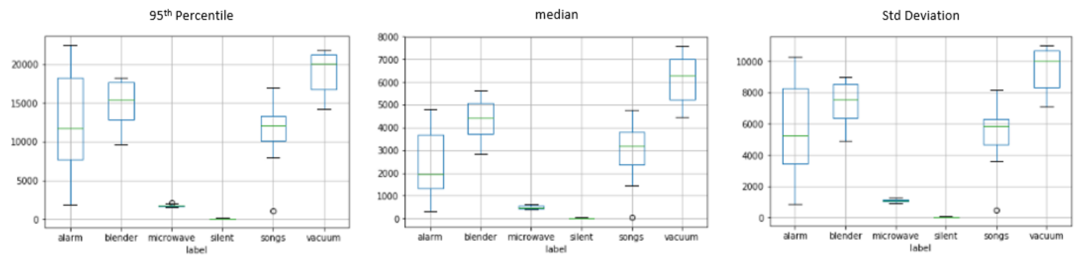
When using spectrograms for feature extraction. I would use the SciPy package to create a spectrogram of either the full clip or a 3 second window. Then I would take the log of the spectrogram pixels (add 1 to avoid log(0) error). I found this gave the picture more definition as it reduced the distance between the highest pixel values and the rest. Also, from my readings and experience I've found this is standard practice. Along with the rescale, I also eliminated frequencies above 10 KHz. As you can see from the exploratory data analysis notebook there is very little information at this level. After formatting the spectrograms, I used OpenCV to resize the array to a set number of time and frequency bins. My bin counts were 50 frequency bins and either 3 or 10 time bins (depending on window or full clip analysis) during the resize.



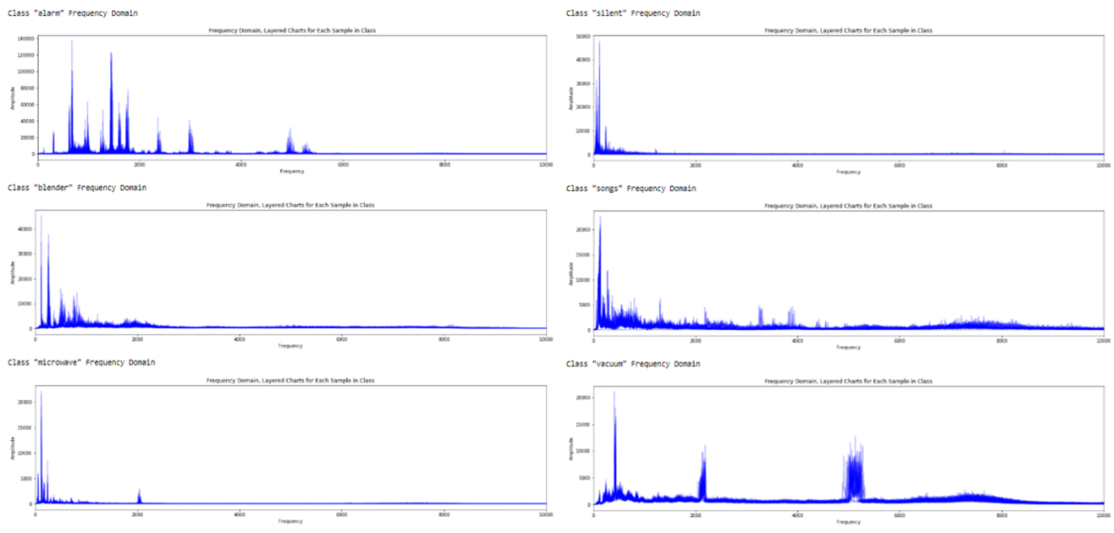
Domain Specific Features:

For my domain feature pipeline, I used a mix of frequency peaks and distributional statistics. The exploratory data analysis notebook details my efforts to chart and select features from a list of candidates.

The distributional statics I chose we rather basic. I used the median and 95th percentile of the absolute value of the audio amplitudes, along with the standard deviation. You can see box plot comparisons of these features across audio classes below. As you'll notice, there is quite a bit of variation in these features across the audio classes.



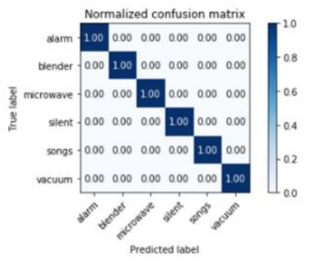
The next set of features I used were from the frequency transform of the audio signals. During the exploratory data analysis step, I found that each of the sound classes had vastly different frequency profiles. The figure below uses overlapping transparent charts to show the frequency profile for each class across all the samples. To summarize the differences between these 6 charts, I used frequency peaks and amplitudes of the top 10 most "prominent" peaks, the peak count, and the median peak frequency. I found that some signal smoothing was required on the Fast Fourier Transform in order to isolate peaks. To do this, I binned the transformed signal using the max value of all amplitudes for each 10 Hz.



RESULTS:

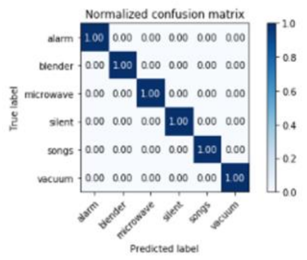
I achieved a 100% classification rate for each of my windowing approaches on the "unseen" test dataset. This accuracy is from aggregating the individual window predictions into 1 final prediction for the full audio clip. The prediction accuracy for each of the individual windows was slightly lower.

Spectrogram Windows



	alarm	blender	microwave	silent	songs	vacuum
alarm	1.00	0.00	0.00	0.00	0.00	0.00
blender	0.00	1.00	0.00	0.00	0.00	0.00
microwave	0.00	0.00	1.00	0.00	0.00	0.00
silent	0.00	0.00	0.00	1.00	0.00	0.00
songs	0.00	0.00	0.00	0.00	1.00	0.00
vacuum	0.00	0.00	0.00	0.00	0.00	1.00

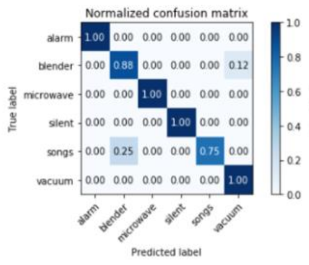
Domain Windows



	alarm	blender	microwave	silent	songs	vacuum
alarm	1.00	0.00	0.00	0.00	0.00	0.00
blender	0.00	1.00	0.00	0.00	0.00	0.00
microwave	0.00	0.00	1.00	0.00	0.00	0.00
silent	0.00	0.00	0.00	1.00	0.00	0.00
songs	0.00	0.00	0.00	0.00	1.00	0.00
vacuum	0.00	0.00	0.00	0.00	0.00	1.00

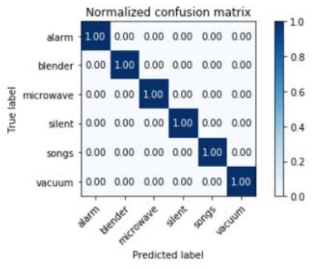
My full audio file prediction accuracy using spectrograms was slightly lower, at around 93% across classes. From the confusion matrix, you can see that all the test error occurs when attempting to classify “blender” and “songs” audio files. The “blender” gets misidentified as a “vacuum” 12% of the time and the “songs” get misidentified as a “blender” 25% of the time. On the other hand, my full audio classification using domain features achieved 100% on the test dataset across all the classes.

Spectrogram Full



	alarm	blender	microwave	silent	songs	vacuum
alarm	1.00	0.00	0.00	0.00	0.00	0.00
blender	0.00	0.88	0.00	0.00	0.00	0.12
microwave	0.00	0.00	1.00	0.00	0.00	0.00
silent	0.00	0.00	0.00	1.00	0.00	0.00
songs	0.00	0.25	0.00	0.00	0.75	0.00
vacuum	0.00	0.00	0.00	0.00	0.00	1.00

Domain, Full



	alarm	blender	microwave	silent	songs	vacuum
alarm	1.00	0.00	0.00	0.00	0.00	0.00
blender	0.00	1.00	0.00	0.00	0.00	0.00
microwave	0.00	0.00	1.00	0.00	0.00	0.00
silent	0.00	0.00	0.00	1.00	0.00	0.00
songs	0.00	0.00	0.00	0.00	1.00	0.00
vacuum	0.00	0.00	0.00	0.00	0.00	1.00

RESOURCES:

Below is a list of resources I consulted while working on this project.

Creating Domain Features:

http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.997.2572&rep=rep1&type=pdf

http://ataspinar.com/2018/04/04/machine-learning-with-signal-processing-techniques/

https://github.com/yoavram/DataSciPy/blob/fa3f53c002dfeb676fe3765c0d7b61ec79efedee/sessions/audio.ipynb

Confusion Matrix Visualization Code

https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html

General Help

https://stackoverflow.com/