ASSIGNMENT-2
Report for Team-3

Purnachandra Gajjala, Cameron Adkins, Gabriel Abeyie

The main goal of this project is to develop a machine learning pipeline that can detect activities and events using sound. The project involved:

(i) Data collection.

(ii) Data pre-processing/signal conditioning.

(iii) Feature extraction, using an existing ML implementation.

(iv) Analysis of results.

Data Collection:

The data collection process for this assignment involved recording 30 seconds samples for each of the 5 classes (plus a 6^{th} class for pure silence) The samples were recorded as PCM at 44.1 kHz in mono. For classes that required equipment that the group did not own, we simply recorded samples from internet sources. Below summarizes this information:

1. Alarm – Internet Source
2. Blender – Internet Source
3. Microwave – Live Recording
4. Music – Live Recording
5. Vacuum – Live Recording
6. Silence – Live Recording

Data Pre-processing / Signal Conditioning:
The collected data was then processed before feature extraction. The first preprocessing step involved pushing the signal through a digital low pass filter (LPF). This LPF's -3db point was set to 15 kHz, as upon inspection many of the signals had no useful information beyond this point. We computed the Fast Fourier Transform (FFT) to convert the filtered digital time-domain signal to a frequency-domain representation. We additionally compute spectrograms for every signal. See the attached figure at the end for a sampling of these outputs.

Feature Selection:
In addition to the standard FFT and spectrogram approaches, we also pulled out some domain specific features for our model:

- ***Signal to Noise Ratio:*** As the name implies this gives us a single number representing signal integrity. This worked fairly well for classes like music or alarm, but falls apart when we try and apply it to classes like microwave or vacuum.
- ***Mel-Freq Cepstral (MFC):*** This transform provides the short-term power spectrum of the signal. Frequency bands in this transform more closely approximate the cochlea in humans which provides better support for mid-range frequencies. We take the mean of this transform to arrive at a single value feature.
- ***Spectral Bandwidth:*** This provides us with which frequencies are at play in the signal. We simply take the mean of these to get a single feature for our vector.

Results:
We split our data in 70% for training and 30% for testing. We adopted a **Random Forest Classification** approach.

Spectrogram Approach

For non-windowing approach,
(a) We conducted a 10-fold Cross Validation test and obtained an average score of **85.7%**.
(b) We also obtained a testing accuracy of **72.2%.**
(c) **Notes:** Due to new microwave samples being obtained late into the project, accuracy for this one has degraded from it's all time best value. This is likely due to the LPF cutting out the higher frequencies from these new samples on the longer timescale. In the current iteration, microwave is not even classified once.

For windowing approach,

(a) We conducted a 10-fold Cross Validation test and obtained an average score of **100%**.
(b) We also obtained a testing accuracy of **100%.**
(c) **Notes:** Shorter windows for the domain appear to help the spectrogram accuracy.

Domain Specific approach

For non-windowing approach,

(d) We conducted a 10-fold Cross Validation test and obtained an average score of **100%**.
(e) We also obtained a testing accuracy of **100%.**
(f) **Notes:** Even with just 3 (rather basic) features, we were able to get very high accuracy from these tests. However, these features can take a very long time to calculate and as such are not as desirable in a live setting.

For windowing approach,

(d) We conducted a 10-fold Cross Validation test and obtained an average score of **100%**.
(e) We also obtained a testing accuracy of **100%.**

Conclusions:

Overall, the train test/split results speak for themselves – this is a very accurate way to classify everyday audio. Unfortunately, this success does not correlate to a live model as well as we had hoped. As mentioned, the features we selected were not expedient computationally which does not lend it self well to the live classification task. Furthermore, a number of bizarre anomalies cropped up while analyzing live data with concern to spectrum magnitude. More work would have to be done to get this to work in a live setting than we currently have time to accomplish, so we leave this as a future endeavor. Code is included in the attached zip for anyone curious.

And the promised spectrograms for our samples: