Introduction To ML
Project Report
Audio Classification using Neural Networks

Sahil Chitnis — N10986573 — ssc9983

21st May 2021

1 Introduction

Audio is one of the 5 sensory parameters that serves as information to humans that helps to understand our surroundings. Thus it is important to be able to differentiate between different sounds to be able to understand the audio source better. Audio has many features that help to differentiate between two sounds but the 3 most important ones are :

a) Amplitude (ie) Loudness of the sound

b) Frequency (ie) The pitch of the sound

c) Timbre (ie) Quality of sound or source of the sound (ex differentiate between dog's bark and cat's meow)

However there are certain attributes that can make differentiation of 2 sounds difficult. Such as,

a) Noise in recorded sounds clips — transducer/background noise

b) Limited Samples of data

c) Uneven sampling rate

Let us see how can we use the features from above, try to remove some of these problematic attributes and classify sounds.

Problem Statement : Classify daily life activities from their audio sounds. Some examples of daily life activities that we are trying to classify correctly:

a) Phone call

b) Clapping

c) Water dropping

d) Sweeping

e) Washing Hands

f) Watching TV

g) Entering / Exiting ie Door opens / closes

h) other — None of the above events

Audio data that i have trained my model on and used to test and validate my model is from University of Moratuwa. Google drive link to dataset . The audio clips recorded are .WAV files.

2 Audio Preprocessing

The Librosa library in python is very useful when it comes to performing functions/processing with audio. Librosa load function allows WAV files to be loaded as numpy arrays. This array contains the various amplitudes of the loaded audio clip at a sampling rate of 22050 or 44100.

a) **Data augmentation** : Since the data used for training is less we can expand it by augmenting the audio data. Data augmentation is the process of creating new samples by modifying (ex time shifting, time-scale modification, pitch shifting,noise addition, and volume control) the existing data. Here i have time shifted the data by shifting the signal to the left/right by some percent. We can see below the effect of Data Augmentation.

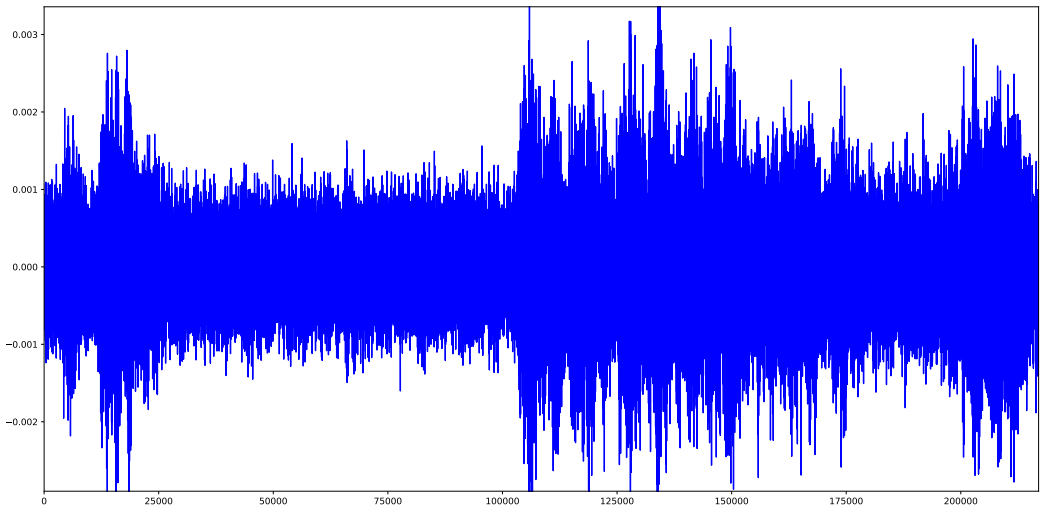

Figure 1: Before Data Processing

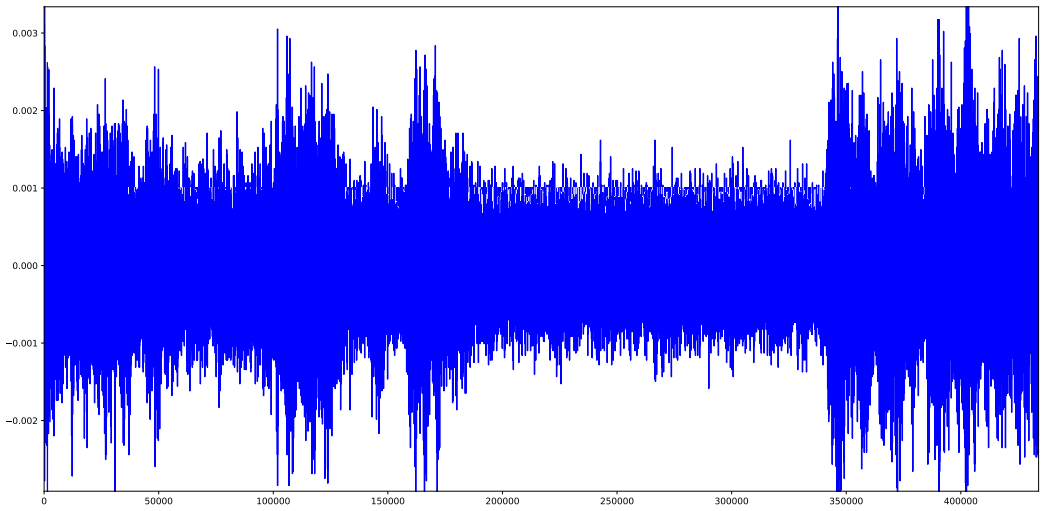

Figure 2: After Data Augmentation

b) **Noise Removal** : As discussed above, the issue we face with audio classification is noise. Thus first i have tried to remove noise. Noisereduce python library helps to accomplish this. Below we can see the affect of reducing noise in one of the samples.

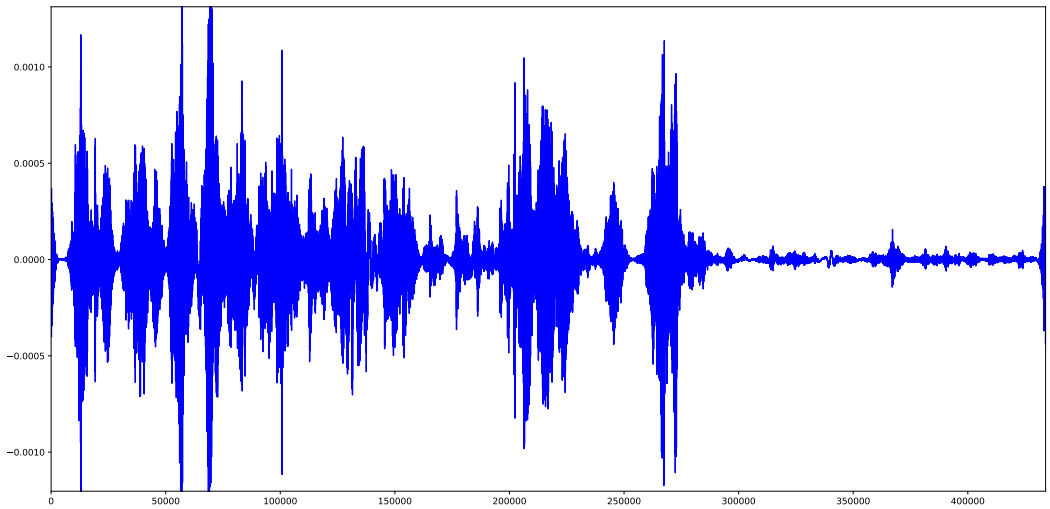

Figure 3: After Noise Reduction

c) **Trimming** : But we can observe that the resulting audio clip has some silence at the start and end and this needs to thus be trimmed. Librosa library's trim functions help to trim these silences. We can see the effect of trimming below

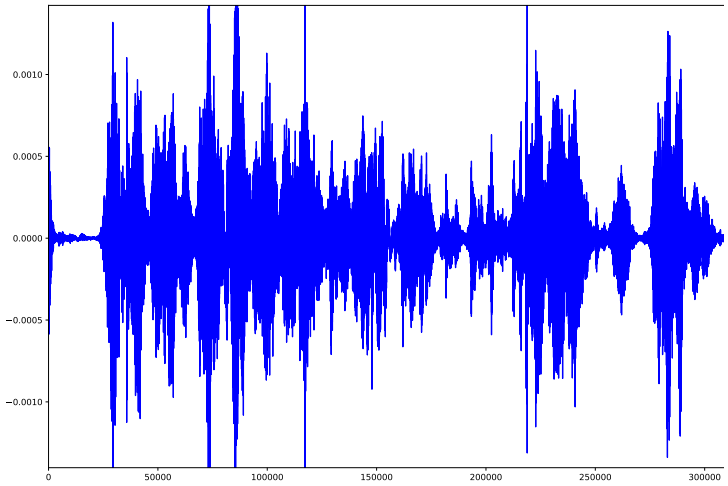

Figure 4: After Trimming

3 Feature Extraction

As mentioned earlier audio classification requires use of certain features to correctly classify the data. **Short time fourier transform (STFT)** is used to determine the sinusoidal frequency and phase content of local sections of a signal as it changes over time. Using this concept we can find features for our pre-processed audio data. Librosa has functionality to calculate STFT from each audio clip. I have used fast fourier transform windows size as 512 and each window has been moved by a jump of 256 to fit a better overlapping. Below we can see the extracted features after STFT.

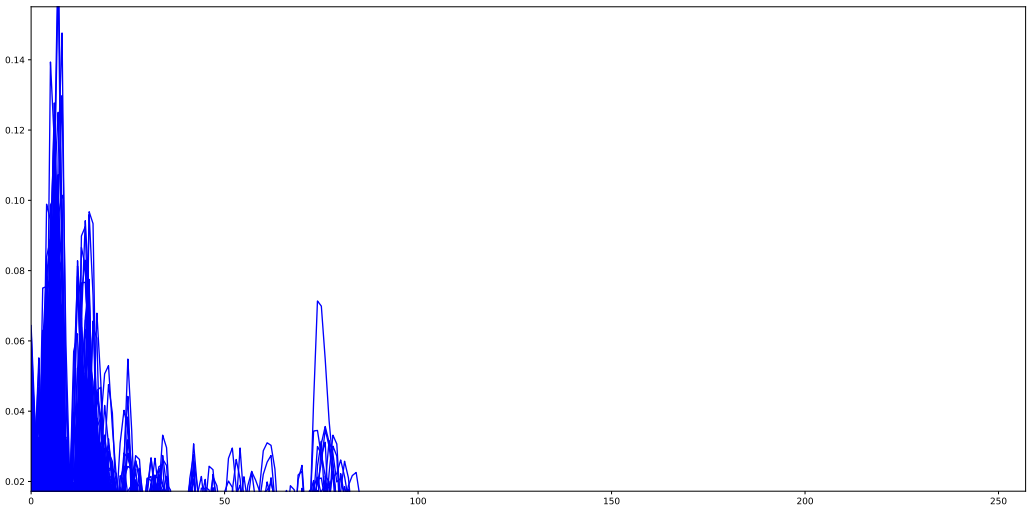

Figure 5: STFT Features

STFT Algorithm Explained:

Let us assume we have N number of samples , window length L and window's jump length J. The STFT algorithm calculates a series of windows received by sliding a fixed length L window by a step of J. Thus total number of windows are 1+(N-L)/J. For each window, the amplitudes of frequency buckets (in Hz) in the range 0 to sampling_rate/2 are recorded. The frequency range is equally spaced when determining the values of the frequency buckets. Thus STFT features for an audio clip will be a 2D array.

However each sound activity can have different number of samples and it is best to flatten the data, in my case i have reduced it to a fixed value of 257.

Different sounds can have different amplitudes, for example something like Clapping can have higher amplitude than Sleeping. To take care of this we need to normalize all sounds to a common level.

4 Audio Data Classification

After pre-processing the data and feature extraction, its finally time to build a Neural Network to train it on our audio data to be help classify test data. But before we build the algorithm let us split the sounds from different activities into Training, test and validation data. Now we fit a 6 layer Neural network on the training data. Layers 4,5 have dropout to reduce overfitting.

a) 1st layer : Dense with o/p = 256 neurons, i/p = 257and RELU activation

b) 2nd layer : Dense with o/p = 256 neurons and RELU activation

c) 3rd layer : Dense with o/p = 128 neurons and RELU activation

d) 4th layer : Dense with o/p = 128 neurons, RELU activation and Dropout = 0.5

d) 5th layer : Dense with o/p = 128 neurons, RELU activation and Dropout = 0.5

e) 6th layer : Dense with o/p = 8 neurons (ie = `num_labels`), Softmax activation

I have used Adam optimizer with a learning rate of 3e-3 after experimenting over different learning rates.

The NN model is trained with 880 samples, validated on 146 samples. We can see an approx accuracy of about **91-93%**. I have also plotted a confusion matrix to show the prediction for different activities.

5 Future Scope : How to improve PREDICTION

a) By visualizing the data or understanding audio science and signal processing better we can figure out better arguments for the STFT function such as better number of frequency buckets, window length L and window's jump length J.

b) We can per-process the data better by empirically selecting better parameters for the noise reduction function.

c) Data augmentation can be improved by augmenting also over pitch (ie pitch shifting)

d) Model needs to be improved by selecting better number of NN layers, activation functions, dropout layers.

6 References

1) pliugithub

2) https://towardsdatascience.com/audio-deep-learning-made-simple-sound-classification-step-by-step-cebc936bbe5

3) https://ai.googleblog.com/2021/03/leaf-learnable-frontend-for-audio.html

4) https://mikesmales.medium.com/sound-classification-using-deep-learning-8bc2aa1990b7

5) https://librosa.org/doc/latest/index.html

6) https://drive.google.com/drive/folders/1s1-174qlu_ekiKcGXeutP5tvrVdjmsQF