PyTorch Audio Emotion Classifier Project write-up:

Audio classification can be used to interpret audio scenario, which is critical in turn for an artificial entity to understand and communicate more efficiently with its environment. This project is a successful implementation of simple Convolutional Neural Networks in pytorch which can predict the emotion of random audio file with 60 percent accuracy. There is a scope to improve the performance by building a robust model and by augmentation of audio files by noise injection, time shift or stretching and pitch shifting.

Dataset Preparation:

I have created a classifier of audio emotions like anger, disgust, sad, joy, fear, surprise and neutral including gender categorizing. My primary goal is to have a dataset which comprises of audio files with above mentioned emotions of different gender and age group. Prepared a dataset with ~12500 audio files merging four different data files by labelling them into 7 different emotions of male and female (clearly explained below).

1. Surrey Audio-Visual Expressed Emotion (SAVEE) Source

Description - Surrey Audio-Visual Expressed Emotion (SAVEE) database has been recorded as a pre-requisite for the development of an automatic emotion recognition system. The database consists of recordings from 4 male actors in 7 different emotions, 480 British English utterances in total. The naming of the audio files is such that the prefix letters identify the groups of emotions as follows: 'a' = 'anger' 'd' = 'disgust' 'f' = 'fear' 'h' = 'happiness' 'n' = 'neutral' 'sa' = 'sadness' 'su' = 'surprise'

2. Toronto emotional speech set (TESS) Source

Description -The Northwestern University Auditory Test No. 6 (NU-6; Tillman & Carhart, 1966) was the model of these stimuli. A set of 200 target words were spoken in the carrier phrase 'Say the word _ ' by two actresses (aged 26 and 64) and recordings were made of the series representing each of the seven emotions (anger, disgust, fear, joy, pleasant surprise, sadness, and neutral), with a total of 2800 stimuli.

***3.Crowd-Sourced Emotional Multimodal Actors (CREMA-D*)** Source

Description - The data set consists of emotional facial and vocal expressions in sentences that are spoken in a variety of specific emotional states (happy, sad, rage, fear, disgust, and neutral). Multiple raters have rated 7,442 clips of 91 actors of diverse ethnic backgrounds in three modalities: audio, visual and audio-visual.

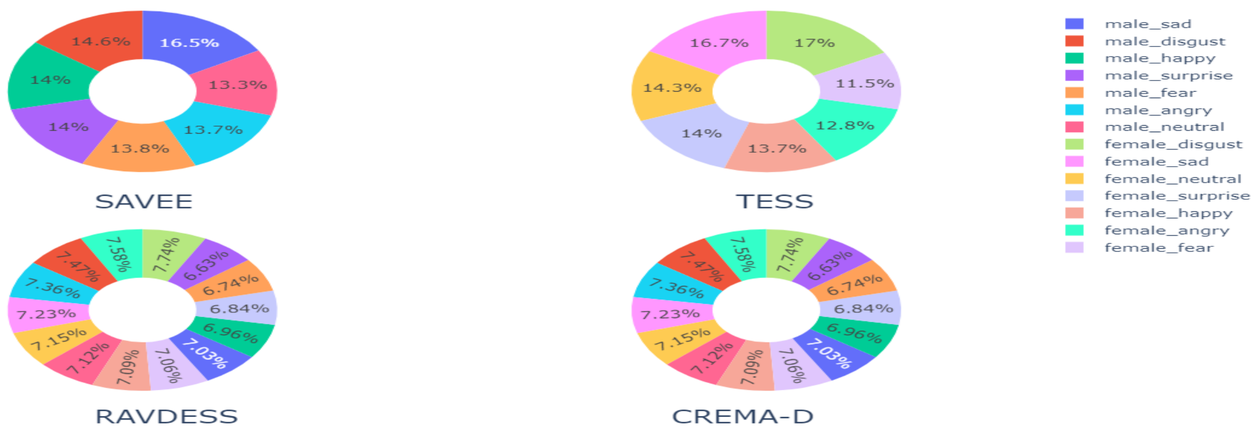
4. The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS)Source

Description - The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS) contains 7356 files. The database contains 24 professional actors (12 females, 12 males), vocalizing two lexically matched statements in a neutral North American accent. Speech includes calm, happy, sad, angry, fearful, surprise, and disgust expressions, and song contains calm, happy, sad, angry, and fearful emotions. All conditions are available in three modality formats: Audio-only (16bit, 48kHz .wav), Audio-Video (720p H.264, AAC 48kHz, .mp4), and Video-only (no sound).

Label Encoding: Using torchnlp.encoders class to get Label Encoder module and encoded each class. So, class can be used as a target variable to train the neural network for predictions on audio file.

Feature extraction: This includes loading the audio file using torch libraries and transform the waveform generated in to 2D or 3D image array on which CNN model could train and predict the emotion of random audio input.

Class Distribution in each Dataset:



PyTorch have an extensive set of libraries which makes our life easy to apply Resampling, MFCC and Padding on the waveform generated from loading the audio file using torchaudio. As you know, 'sound' is an analog signal, to make it digital and able to represent it with something like a numpy array we must sample the original signal. Resampling or down sampling helps us reduce or compress the size of audio file by 1/10th. As I am working with different audio files it is a good idea to sample them all to one sample rate.

The idea of MFCC is to convert audio in time domain(analog) into frequency domain so that we can understand all the information present in speech signals. But just converting time domain signals into frequency domain may not be very optimal. We can do more than just converting time domain signals into frequency domain signals. Our ear has cochlea which basically has more filters at low frequency and very few filters at higher frequency. This can be mimicked using Mel filters. So, the idea of MFCC is to convert time domain signals into frequency domain signal by mimicking cochlea function using Mel filters. Padding is done to set length of all torch tensors to same length.

The Audio Dataset class loads all the audio files and returns Audio tensors and Audio labels of each file as an input to CNN model after giving input to pytorch data loader.

Predictions:
Crossentropy loss, measures the performance of a classification model whose output is a probability value between 0 and 1, loss increases as the predicted probability diverges from the actual label. Simple CNN model embedded with couple of pooling layers and Batch-norm gives 62 percent accuracy on training set and 55 percent accuracy on validation set. Upon testing the model with random audio files model predicted right on clear audio files but failed to predict a tricky audio file.

Conclusions: Successful implementation of neural nets to predict emotion of audio files with 55% accuracy using MFCC feature extraction method. Building a robust model and applying audio augmentation would have helped in gaining better prediction accuracy.